DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

Infraestructura Computacional 202510 – <Caso 3 >

Apellidos	Nombres	Código	Login
Chaparro Diaz	Andrés Felipe	202111146	a.chaparrod
Sánchez Novoa	Edward Camilo	202113020	e.sanchezn

Tabla de Contenido

1. Introducción
2. Objetivos
3. Problemática
4. Implementación del Prototipo
   4.1. Organización de Archivos
   4.2. Descripción de los Componentes Clave
   4.3. Instrucciones de Ejecución
5. Metodología para Mediciones
   5.1. Escenarios Evaluados
   5.2. Mediciones Realizadas
   5.3. Proceso de Recolección de Datos
6. Resultados de las Mediciones
   6.1. Tabla de Datos Recopilados (Promedios)
   6.2. Gráficas de Rendimiento
   6.2.1. Tiempos de Firma vs. Escenarios (Gráfica 1)
   6.2.2. Tiempos de Cifrado Simétrico de la Tabla vs. Escenarios (Gráfica 2)
   6.2.3. Tiempos de Verificación de la Consulta (HMAC) vs. Escenarios (Gráfica 3)
   6.2.4. Tiempos de Cifrado Simétrico vs. Asimétrico (Servidor) vs. Escenarios (Gráfica 4)
7. Análisis y Discusión de los Resultados
   7.1. Comentarios sobre la Gráfica 1: Tiempos de Firma
   7.2. Comentarios sobre la Gráfica 2: Tiempos de Cifrado Simétrico
   7.3. Comentarios sobre la Gráfica 3: Tiempos de Verificación de la Consulta
   7.4. Comentarios sobre la Gráfica 4: Comparación Cifrado Simétrico vs. Asimétrico
   7.5. Ventajas y Limitaciones de los Algoritmos
8. Estimación de la Velocidad del Procesador
   8.1. Escenario Elegido
   8.2. Cálculos para Cifrado Simétrico (AES)
   8.3. Cálculos para Cifrado Asimétrico (RSA)
9. Conclusiones
10. Referencias

1. Introducción

Este informe muestra el desarrollo e implementación paso a paso de un prototipo a escala para un sistema de comunicación segura entre un cliente y un servidor principal, enfocado en garantizar la confidencialidad e integridad de la información intercambiada. El proyecto se basa en un caso de estudio que simula los servicios de consulta en línea de una aerolínea, donde el cliente interactúa inicialmente con un servidor principal para obtener información sobre servidores delegados que atienden servicios específicos. Se han implementado diversos algoritmos criptográficos estándar en Java para asegurar las comunicaciones, incluyendo Diffie-Hellman para el intercambio de llaves de sesión, AES para cifrado simétrico, RSA para la firma digital, y HMAC-SHA256 para la verificación de integridad y autenticidad. Además de la implementación funcional, se ha realizado un análisis de rendimiento para evaluar la sobrecarga computacional introducida por estos mecanismos de seguridad en diferentes escenarios de carga (secuencial y concurrente), comparando los tiempos de operaciones clave como firma, cifrado y verificación.

2. Objetivos

De acuerdo con el enunciado del caso de estudio, los objetivos del proyecto son:

- Comprender las ventajas y limitaciones de los algoritmos para cifrar datos (simétricos y asimétricos), firma digital y funciones hash con llave (HMAC).
- Construir un prototipo a escala de una herramienta en Java que soporte confidencialidad e integridad en las comunicaciones entre cliente y servidor principal, utilizando sockets y algoritmos criptográficos estándar sin librerías especiales más allá de java.security y javax.crypto.
- Generar y utilizar llaves RSA preexistentes (pública y privada) para el servidor, y solo la llave pública para el cliente.
- Implementar el protocolo de comunicación descrito en el caso de estudio, incluyendo el intercambio de llaves de sesión mediante Diffie-Hellman, la derivación de llaves de cifrado (AES) y MAC (HMAC) a partir de un secreto compartido, y el uso de estos para cifrar, descifrar, firmar, verificar y calcular HMAC sobre los datos intercambiados.
- Medir el tiempo requerido por el servidor para operaciones específicas (firmar, cifrar la tabla, verificar la consulta) en escenarios secuenciales y concurrentes.
- Comparar el tiempo de cifrado simétrico con un cálculo auxiliar de cifrado asimétrico de la respuesta del servidor.
- Analizar los datos de rendimiento recopilados y estimar la velocidad del procesador en términos de operaciones criptográficas por segundo.

3. Problemática

La problemática central aborda la necesidad de asegurar la comunicación entre un cliente y el servidor principal de una aerolínea. Aunque la aerolínea distribuye sus servicios en servidores delegados, la interacción inicial para descubrir estos servicios y establecer una

comunicación segura se realiza con un servidor principal. Esta comunicación inicial debe garantizar:

- **Confidencialidad:** Que la información intercambiada (identificadores de servicio, direcciones IP/puertos) no pueda ser interceptada y leída por terceros no autorizados.
- **Integridad:** Que los datos recibidos no hayan sido alterados durante la transmisión.
- **Autenticidad:** Que el cliente pueda verificar que la información recibida (particularmente la tabla de servicios inicial) proviene realmente del servidor principal legítimo, y que el servidor principal pueda verificar que las solicitudes (identificador de servicio) provienen del cliente con el que estableció la sesión segura.

El desafío radica en implementar estos mecanismos utilizando únicamente librerías estándar de Java, configurando un protocolo de intercambio de llaves seguro (Diffie-Hellman híbrido con firma RSA) y gestionando la sobrecarga computacional que introducen estos algoritmos, especialmente en escenarios de alta concurrencia. La implementación debe reflejar el protocolo especificado, donde el cliente recibe la tabla de servicios cifrada y firmada, selecciona un servicio, y luego solicita y recibe la dirección IP/puerto del servidor delegado correspondiente, también de forma segura.

4. Implementación del Prototipo

La implementación del prototipo se realizó en Java, utilizando exclusivamente las APIs estándar java.security y javax.crypto. El código se organiza en varios archivos dentro de un paquete src.crypto.

4.1. Organización de Archivos

El contenido del archivo zip de entrega (.zip) la sigutiene iente estructura:

```
├── Caso3
│   ├── Caso3
│   │   ├── keys
│   │   │   ├── private.key    (Llave privada RSA del servidor - binaria)
│   │   │   └── public.key     (Llave pública RSA del servidor - binaria)
│   │   ├── src
│   │   │   └── crypto
│   │   │       ├── Cifrado.java             (Clase con métodos para cifrado, firma, HMAC)
│   │   │       ├── Cliente.java             (Cliente estándar para ejecución manual)
│   │   │       ├── ClienteMedicion.java     (Cliente modificado para mediciones automáticas)
│   │   │       ├── DelegadoServidor.java    (Maneja la conexión individual de cada cliente en el servidor)
│   │   │       ├── DiffieHellman.java       (Clase con métodos para el intercambio Diffie-Hellman)
│   │   │       ├── GeneradorLlavesRSA.java  (Utility para generar las llaves RSA)
│   │   │       ├── Main.java                (Posible punto de entrada para ejecutar escenarios)
```

```
│   │   │       ├── MedicionesMain.java      (Clase principal para ejecutar los escenarios de medición
y generar CSV)
│   │   │       ├── MedidorTiempo.java       (Clase auxiliar para medir tiempos de ejecución)
│   │   │       └── Servidor.java            (Servidor principal)
│   ├── docs
│   │   └── informe.pdf          (O .docx, .md, etc. - Este documento)
│   └── mediciones.csv           (Archivo con los datos brutos de las mediciones)
└── README.md                    (Breve descripción y cómo iniciar)
```

4.2. Descripción de los Componentes Clave

- GeneradorLlavesRSA.java: Un programa simple para crear un par de llaves RSA (pública y privada) de 1024 bits y guardarlas en archivos binarios (public.key, private.key) en el directorio keys. Es el primer paso antes de ejecutar el servidor o cliente.
- Cifrado.java: Contiene métodos estáticos para realizar las operaciones criptográficas requeridas: cifrado/descifrado AES (CBC, PKCS5Padding, 256 bits), cifrado/descifrado RSA (1024 bits), firma/verificación SHA256withRSA, cálculo de HMAC-SHA256 y generación de IVs aleatorios. También incluye métodos helper para crear SecretKey a partir de bytes.
- DiffieHellman.java: Encapsula la lógica para el intercambio Diffie-Hellman de 1024 bits, incluyendo la generación de parámetros, la creación de pares de llaves DH, la realización de la fase de acuerdo y la derivación del secreto compartido.
- Servidor.java: Representa el servidor principal. Carga sus llaves RSA al iniciar. Mantiene tablas estáticas de servicios y sus direcciones asociadas. Escucha en un puerto específico y, por cada conexión de cliente entrante, crea una nueva instancia de DelegadoServidor para manejar la sesión en un hilo separado. Esto permite la concurrencia. Incluye un flag ejecutando para permitir un cierre ordenado.
- DelegadoServidor.java: Implementa la lógica de comunicación segura con un *único* cliente. Ejecuta el protocolo Diffie-Hellman para establecer las llaves de sesión simétricas. Envía la tabla de servicios (serializada como texto), cifrada con AES, firmada con la llave privada RSA del servidor y con un HMAC. Recibe la solicitud de servicio del cliente (un ID), la descifra, verifica su HMAC, busca la dirección IP/puerto correspondiente y se la envía al cliente, cifrada con AES y con un HMAC. Esta clase incluye instancias de MedidorTiempo para registrar los tiempos de operaciones específicas en el servidor (firma de la tabla, cifrado simétrico/asimétrico de la respuesta, verificación del HMAC de la solicitud).
- Cliente.java: Implementa la lógica del cliente estándar para interacción manual. Carga la llave pública RSA del servidor. Se conecta al servidor, realiza el intercambio Diffie-Hellman para derivar las llaves de sesión, recibe la tabla de servicios (verificando HMAC y firma, luego descifrando), la muestra al usuario, le pide seleccionar un ID, cifra

la solicitud, calcula su HMAC, la envía, recibe la dirección IP/puerto (verificando HMAC, luego descifrando) y la muestra.

- MedidorTiempo.java: Una clase simple para medir la duración entre un inicio y un fin en nanosegundos y convertirlo a milisegundos. Utilizada dentro de DelegadoServidor y ClienteMedicion.
- ClienteMedicion.java: Una versión modificada de Cliente.java diseñada para ser usada en los escenarios de medición. No interactúa con el usuario, sino que siempre solicita el servicio con ID "1" (o un ID fijo). Integra MedidorTiempo para registrar tiempos en el lado del cliente para todas las operaciones criptográficas, y reporta estos tiempos (incluyendo las mediciones del servidor que recibe) a un mecanismo centralizado (MedicionesMain) para guardarlos en el CSV. Recibe un ID para diferenciar las entradas en el CSV.
- MedicionesMain.java: La clase principal encargada de orquestar los escenarios de medición. Inicializa el archivo mediciones.csv. Ejecuta el escenario iterativo (iniciando un servidor y un cliente secuencial que hace 32 consultas) y los escenarios concurrentes (iniciando un servidor y N clientes concurrentes para N=4, 16, 32, 64). Utiliza un lock para asegurar la escritura sincronizada en el archivo CSV desde múltiples hilos/clientes/delegados.

4.3. Instrucciones de Ejecución

Para compilar y ejecutar el prototipo:

1. **Asegurar entorno:** Tener instalado Java Development Kit (JDK) compatible con Java 8 o superior (para asegurar la disponibilidad de las APIs criptográficas estándar).
2. **Compilar:** Navegar al directorio Caso3/Caso3/src/crypto en la terminal y compilar todos los archivos Java:

   javac *.java

3. **Generar Llaves RSA:** Si es la primera vez que se ejecuta, o si se desean generar nuevas llaves, ejecutar la utilidad de generación de llaves. Esto creará el directorio keys y los archivos public.key y private.key dentro de Caso3/Caso3/keys.

java src.crypto.GeneradorLlavesRSA

content_copydownload

4. **Ejecución para Mediciones:**
   - Navegar al directorio Caso3/Caso3/src/crypto en la terminal.
   - Ejecutar la clase MedicionesMain. Esta clase se encargará automáticamente de iniciar el servidor y los clientes para cada escenario y registrar los tiempos en mediciones.csv.

   java src.crypto.MedicionesMain

- El programa MedicionesMain ejecutará primero el escenario secuencial (32 consultas) y luego los escenarios concurrentes (4, 16, 32, 64 clientes). Esperar a que termine. El archivo mediciones.csv se creará o actualizará en el directorio raíz del proyecto (Caso3).

5. Metodología para Mediciones

Para evaluar la sobrecarga generada por los algoritmos de cifrado y manejo de integridad, se implementó un sistema de medición utilizando las clases MedidorTiempo, ClienteMedicion, DelegadoServidor y MedicionesMain. Las mediciones se registraron para diversas operaciones criptográficas tanto en el lado del servidor principal (DelegadoServidor) como en el lado del cliente (ClienteMedicion), y se guardaron en el archivo mediciones.csv.

5.1. Escenarios Evaluados

Se evaluaron dos tipos de escenarios principales, conforme a la Tarea 1, para simular diferentes condiciones de carga:

- **Escenario Secuencial:** Un único servidor principal atiende a un único cliente iterativo. El cliente realiza 32 consultas secuenciales consecutivas al servidor principal. Este escenario representa una carga ligera y permite medir el rendimiento de operaciones individuales sin la influencia significativa de la concurrencia.
- **Escenarios Concurrentes:** Un servidor principal atiende simultáneamente a múltiples clientes. Se evaluaron diferentes niveles de concurrencia, variando el número de clientes y servidores delegados concurrentes entre 4, 16, 32 y 64. En cada nivel, se inician N hilos cliente, cada uno realizando una única conexión y una única consulta al servidor. Este escenario representa una carga más pesada y permite observar cómo el sistema (y el rendimiento de las operaciones criptográficas por hilo) se comporta bajo presión.

5.2. Operaciones Criptográficas Medidas

Durante la ejecución de los escenarios de medición, se registró el tiempo de diversas operaciones criptográficas clave tanto en el servidor (dentro de cada DelegadoServidor) como en el cliente (ClienteMedicion). Los tiempos registrados en el archivo mediciones.csv (en milisegundos) corresponden a las siguientes operaciones:

- **Operaciones Medidas en el Servidor Principal (DelegadoServidor):**
  - firma: Tiempo que tarda el servidor en firmar digitalmente la tabla de servicios usando RSA. Esta operación usa la llave privada del servidor para crear una firma digital que garantiza la autenticidad de la información.
  - cifrado_simetrico_servidor: Tiempo que tarda el servidor en cifrar datos (la tabla de servicios o la dirección) utilizando el algoritmo AES en modo CBC. Este cifrado es rápido y eficiente para grandes volúmenes de datos.

- verificacion_servidor: Tiempo que tarda el servidor en verificar el HMAC de una solicitud del cliente (la solicitud de servicio o la dirección). Esto asegura la integridad de los datos recibidos del cliente.
- cifrado_asimetrico_servidor: Tiempo que tarda el servidor en cifrar datos (la dirección) utilizando RSA. Esta medición se realiza auxiliarmente para comparación con el cifrado simétrico.

- **Operaciones Medidas en el Cliente (ClienteMedicion):**
  - cifrado_asimetrico: Tiempo que tarda el cliente en cifrar datos (la solicitud) utilizando RSA. Se usa principalmente para cifrar datos pequeños o claves. Medición auxiliar.
  - verificacion: Tiempo que tarda el cliente en verificar la firma digital de la tabla de servicios recibida del servidor. Esta operación confirma que la información proviene realmente del servidor legítimo usando la llave pública del servidor.
  - descifrado_direccion: Tiempo que tarda el cliente en descifrar la dirección recibida del servidor usando AES.
  - hmac_solicitud: Tiempo que tarda el cliente en generar un HMAC para la solicitud que envía al servidor. El HMAC garantiza la integridad de la solicitud.
  - hmac_direccion: Tiempo que tarda el cliente en calcular el HMAC de la dirección recibida del servidor para verificar su integridad.
  - hmac: Tiempo que tarda el cliente en calcular el HMAC de la tabla cifrada para verificar su integridad.
  - descifrado: Tiempo que tarda el cliente en descifrar la tabla de servicios recibida del servidor usando AES.
  - generacion_iv: Tiempo que tarda el cliente en generar un vector de inicialización (IV) aleatorio para el cifrado AES en modo CBC. El IV garantiza que mensajes idénticos produzcan textos cifrados diferentes con la misma llave.
  - cifrado_simetrico: Tiempo que tarda el cliente en cifrar datos (la solicitud) usando el algoritmo AES en modo CBC con padding PKCS5.

5.3. Proceso de Recolección de Datos

La recolección de datos se automatizó con la clase MedicionesMain. Para cada escenario:

- Se inicia una instancia del Servidor.
- Se inician las instancias del ClienteMedicion (ya sea secuencialmente 32 veces para el primer escenario, o N instancias concurrentes para los escenarios concurrentes).

- Cada DelegadoServidor en el servidor y cada ClienteMedicion registran los tiempos de las operaciones criptográficas relevantes utilizando MedidorTiempo.
- Estos tiempos son reportados y escritos, de forma sincronizada, en el archivo mediciones.csv, incluyendo columnas para el escenario, el número total de clientes en ese escenario, un identificador único para cada cliente/consulta, la operación medida y el tiempo en milisegundos.
- Se ejecutó cada escenario una vez con el número especificado de clientes/consultas.

Los datos brutos se encuentran en el archivo mediciones.csv proporcionado. Para la Tarea 2 y las gráficas, se calcularon los tiempos *promedio* para cada operación dentro de cada escenario (Secuencial, Concurrente 4, Concurrente 16, Concurrente 32, Concurrente 64). Por ejemplo, para el escenario "Secuencial", el promedio de "firma" se calcula sobre las 32 entradas de "firma" registradas. Para el escenario "Concurrente, 16", el promedio de "firma" se calcula sobre las 16 entradas de "firma" registradas por los 16 delegados.

6. Resultados de las Mediciones

Los datos brutos recopilados se encuentran en el archivo mediciones.csv. A continuación, se presenta una tabla resumen con los tiempos promedio de las operaciones solicitadas en el servidor para cada escenario evaluado.

6.1. Tabla de Datos Recopilados (Promedios en ms)

Datos Recopilados Escenario Secuencial

	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
1	0,5427	0,1000	0,0782	0,1753	0,0855	0,0846	0,6348	0,0992	0,0342	0,0273	0,0377	0,0983	0,0505
2	0,1556	0,1358	0,1351	0,1978	0,1440	0,1586	0,5384	0,1302	0,0618	0,0488	0,0666	0,1314	0,0630
3	0,1652	0,1306	0,1330	0,2155	0,1475	0,1579	0,6219	0,1304	0,0766	0,0479	0,0813	0,1321	0,0672
4	0,1896	0,1445	0,1614	0,2449	0,1736	0,1948	0,6223	0,1575	0,0794	0,0587	0,0762	0,1483	0,0824
5	0,1189	0,1036	0,1148	0,1761	0,1135	0,1270	0,4416	0,1040	0,0546	0,0365	0,0515	0,1060	0,0484
6	0,1267	0,1009	0,1122	0,1734	0,1161	0,1260	0,4428	0,1104	0,0562	0,0376	0,0597	0,1085	0,0537
7	0,1251	0,1083	0,1175	0,1638	0,1183	0,1285	0,4984	0,1176	0,0600	0,0408	0,0542	0,1116	0,0509
8	0,1202	0,0918	0,1097	0,1596	0,1301	0,1321	0,5022	0,1188	0,0545	0,0383	0,0458	0,1059	0,0487
9	0,1381	0,1074	0,1214	0,1615	0,1291	0,1257	0,5222	0,1072	0,0612	0,0382	0,0520	0,1150	0,0537
10	0,1370	0,1272	0,1242	0,1745	0,1276	0,1373	0,4880	0,1152	0,0511	0,0371	0,0525	0,1201	0,0475
Promedio	0,1819	0,1150	0,1207	0,1842	0,1285	0,1372	0,5313	0,1190	0,0590	0,0411	0,0578	0,1177	0,0566

Datos Recopilados Escenario Concurrencia 4

	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
1	0,0990	0,0849	0,0741	0,1260	0,0852	0,0753	0,3288	0,1080	0,0224	0,0276	0,0440	0,0904	0,0288
2	0,1066	0,0939	0,0927	0,1626	0,0983	0,1180	0,3919	0,1109	0,0260	0,0318	0,0450	0,1022	0,0346
3	0,1184	0,1016	0,0931	0,1790	0,1172	0,1194	0,3948	0,1280	0,0350	0,0337	0,0542	0,1188	0,0634
4	0,1081	0,1067	0,0926	0,1506	0,1060	0,1323	0,4087	0,1442	0,0301	0,0409	0,0645	0,1319	0,0417
5	0,0851	0,0678	0,0601	0,1193	0,0873	0,0824	0,3199	0,0851	0,0258	0,0256	0,0355	0,0935	0,0281
6	0,0923	0,0803	0,0858	0,1364	0,1027	0,1035	0,3510	0,1116	0,0326	0,0291	0,0347	0,0827	0,0324
7	0,0926	0,0928	0,0672	0,1925	0,1077	0,1012	0,3516	0,0955	0,0368	0,0349	0,0408	0,0954	0,0321
8	0,1064	0,0741	0,0950	0,1182	0,0838	0,0873	0,3603	0,0977	0,0249	0,0248	0,0422	0,0948	0,0304
9	0,0966	0,0895	0,0925	0,1501	0,1040	0,1594	0,4849	0,1490	0,0310	0,0300	0,0459	0,0914	0,0402
10	0,0729	0,0652	0,0572	0,1175	0,0675	0,0808	0,2778	0,0867	0,0208	0,0270	0,0415	0,0864	0,0294
Promedio	0,0978	0,0857	0,0810	0,1452	0,0960	0,1059	0,3670	0,1117	0,0285	0,0305	0,0448	0,0988	0,0361

Datos Recopilados Escenario Concurrencia 16

	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
1	0,0885	0,0783	0,0906	0,1962	0,1197	0,0902	0,5399	0,1131	0,0319	0,0285	0,0369	0,0932	0,0395
2	0,0934	0,0853	0,0916	0,2037	0,1106	0,0996	0,4555	0,1180	0,0334	0,0317	0,0459	0,0991	0,0390
3	0,1009	0,0809	0,0827	0,1925	0,1137	0,1155	0,4553	0,1144	0,0298	0,0307	0,0406	0,1017	0,0421
4	0,0958	0,0804	0,0857	0,1868	0,1002	0,0958	0,4913	0,1170	0,0302	0,0382	0,0402	0,0984	0,0423
5	0,0800	0,0751	0,0709	0,1471	0,0978	0,0800	0,4313	0,1335	0,0274	0,0280	0,0317	0,0784	0,0335
6	0,0783	0,0722	0,0889	0,1672	0,0880	0,0766	0,4414	0,1033	0,0272	0,0259	0,0330	0,0807	0,0313
7	0,0754	0,0703	0,0740	0,1711	0,0860	0,0759	0,4384	0,1041	0,0273	0,0255	0,0334	0,0912	0,0341
8	0,0818	0,0697	0,0773	0,1666	0,1002	0,0842	0,4726	0,1011	0,0298	0,0278	0,0354	0,0853	0,0323
9	0,0842	0,0750	0,0834	0,1893	0,0958	0,0902	0,4530	0,1085	0,0304	0,0297	0,0457	0,0862	0,0465
10	0,0801	0,0720	0,0835	0,1829	0,0885	0,0926	0,4597	0,1116	0,0266	0,0282	0,0378	0,0910	0,0344
Promedio	0,0858	0,0759	0,0829	0,1803	0,1000	0,0901	0,4638	0,1124	0,0294	0,0294	0,0381	0,0905	0,0375

Datos Recopilados Escenario Concurrencia 32

	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
1	0,0745	0,0675	0,0720	0,1297	0,0732	0,0663	0,5253	0,0951	0,0268	0,0267	0,0323	0,0889	0,0291
2	0,0813	0,0770	0,0697	0,1475	0,0729	0,0780	0,6046	0,0980	0,0278	0,0265	0,0334	0,0939	0,0322
3	0,0927	0,0889	0,0953	0,1661	0,0849	0,0966	0,5234	0,1134	0,0303	0,0273	0,0423	0,1186	0,0353
4	0,1165	0,1087	0,1040	0,2034	0,1085	0,1019	0,6388	0,1371	0,0372	0,0363	0,0485	0,1238	0,0476
5	0,0769	0,0658	0,0659	0,1230	0,0883	0,0670	0,4822	0,0969	0,0279	0,0252	0,0326	0,0960	0,0291
6	0,0753	0,0684	0,0625	0,1305	0,0669	0,0692	0,5288	0,0930	0,0257	0,0260	0,0291	0,0947	0,0275
7	0,0809	0,0686	0,0682	0,1231	0,0727	0,0706	0,5217	0,1011	0,0254	0,0257	0,0341	0,0993	0,0311
8	0,0750	0,0652	0,0683	0,1313	0,0760	0,0765	0,4883	0,0974	0,0271	0,0252	0,0328	0,0969	0,0314
9	0,0751	0,0679	0,0706	0,1219	0,0702	0,0637	0,4705	0,0974	0,0288	0,0260	0,0320	0,0953	0,0300
10	0,0970	0,0824	0,0987	0,1510	0,0915	0,1007	0,5729	0,1328	0,0313	0,0314	0,0431	0,1083	0,0469
Promedio	0,0845	0,0760	0,0775	0,1427	0,0805	0,0790	0,5356	0,1062	0,0288	0,0276	0,0360	0,1016	0,0340

Datos Recopilados Escenario Concurrencia 64

	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
1	0,0660	0,0606	0,0438	0,0838	0,0522	0,0447	0,5964	0,0646	0,0191	0,0158	0,0211	0,0733	0,0189
2	0,0941	0,0814	0,0609	0,1138	0,0603	0,0629	0,6358	0,0879	0,0401	0,0200	0,0292	0,1018	0,0264
3	0,0811	0,0776	0,0619	0,1229	0,0657	0,0567	0,6166	0,0900	0,0213	0,0210	0,0279	0,0980	0,0244
4	0,0850	0,0750	0,0596	0,1138	0,0657	0,0613	1,3900	0,0916	0,0254	0,0199	0,0283	0,0989	0,0266
5	0,0647	0,0613	0,0421	0,0860	0,0564	0,0468	0,4988	0,0678	0,0186	0,0175	0,0226	0,0732	0,0181
6	0,0647	0,0569	0,0447	0,0863	0,0514	0,0456	0,5179	0,0644	0,0199	0,0144	0,0222	0,0756	0,0220
7	0,0652	0,0600	0,0408	0,0829	0,0486	0,0423	0,7625	0,0681	0,0177	0,0154	0,0197	0,0738	0,0182
8	9,6076	0,0596	0,0443	0,0941	0,0529	0,0453	0,7437	0,0788	0,0198	0,0162	0,0277	0,0768	0,0283
9	0,0727	0,0658	0,0457	0,0909	0,0508	0,0478	0,5110	0,0684	0,0204	0,0167	0,0235	0,0776	0,0216
10	0,0933	0,0799	0,0597	0,1192	0,0662	0,0588	0,6162	0,0958	0,0240	0,0194	0,0302	0,1007	0,0327
Promedio	1,0294	0,0678	0,0504	0,0994	0,0570	0,0512	0,6889	0,0777	0,0226	0,0176	0,0252	0,0850	0,0237

Datos Recopilados Promedio Escenarios

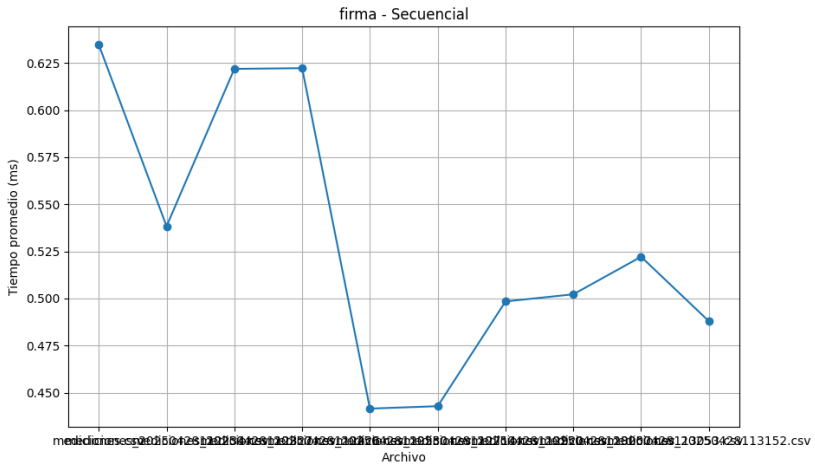
Escenario	cifrado asimetrico	cifrado asimetrico servidor	cifrado simetrico	cifrado simetrico servidor	descifrado	descifrado direccion	firma	generacion iv	hmac	hmac direccion	hmac solicitud	verificacion	verificacion servidor
Secuencial	0,18192	0,11501	0,12074	0,18424	0,12853	0,13724	0,53125	0,11905	0,05896	0,04112	0,05775	0,11771	0,05661
Concurrente 4	0,09778	0,08566	0,08103	0,14520	0,09596	0,10595	0,36696	0,11168	0,02852	0,03051	0,04482	0,09876	0,03610
Concurrente 16	0,08583	0,07593	0,08287	0,18035	0,10004	0,09006	0,46382	0,11245	0,02939	0,02940	0,03805	0,09052	0,03748
Concurrente 32	0,08452	0,07604	0,07751	0,14274	0,08051	0,07905	0,53564	0,10622	0,02883	0,02764	0,03603	0,10156	0,03402
Concurrente 64	1,02943	0,06780	0,05036	0,09936	0,05702	0,05121	0,68888	0,07774	0,02261	0,01765	0,02523	0,08498	0,02372

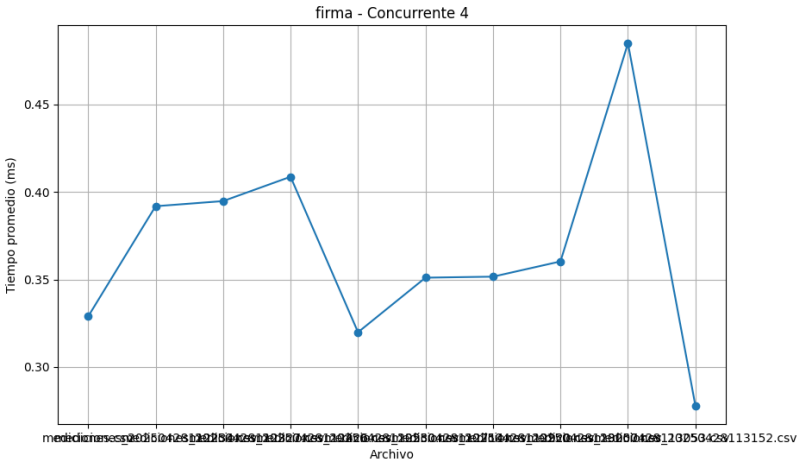
6.2. Gráficas de Rendimiento

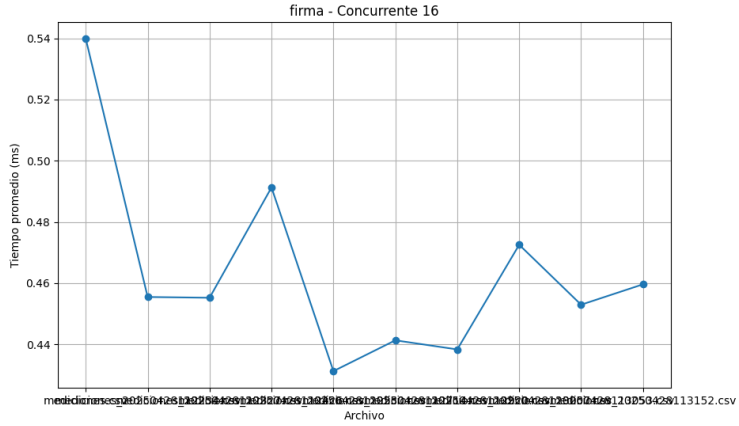
A continuación, se presentarán las gráficas visualizando los datos promedio de la tabla anterior, comparando el rendimiento de las operaciones clave en los diferentes escenarios.

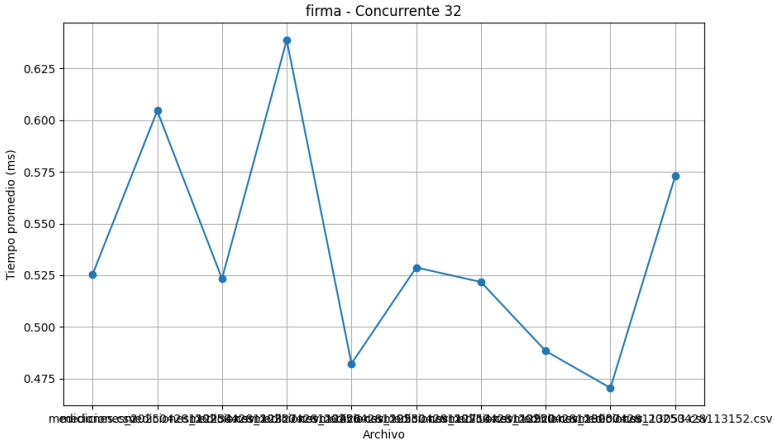
6.2.1. Tiempos de Firma vs. Escenarios

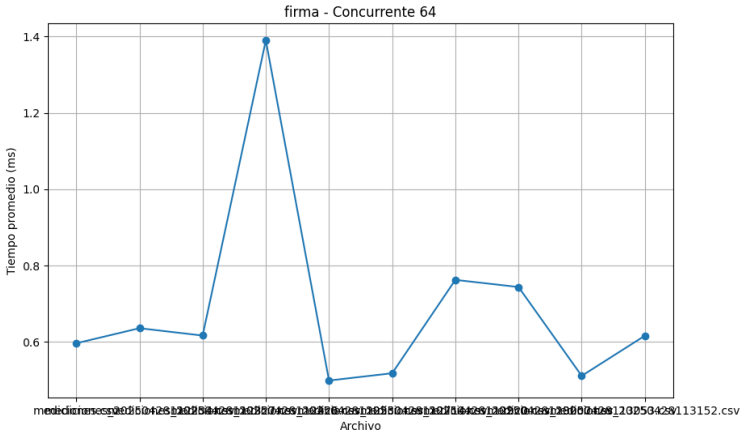
En estas gráficas, se puede observar que la operación de firma digital (firma) realizada por el servidor tiende a mostrar tiempos que, aunque variables en las ejecuciones individuales, se mantienen en un rango relativamente alto en comparación con las operaciones simétricas.




firma - Concurrente 4
Tiempo promedio (ms)
0.45
0.40
0.35
0.30
Archivo


firma - Concurrente 16
Tiempo promedio (ms)
0.54
0.52
0.50
0.48
0.46
0.44
Archivo

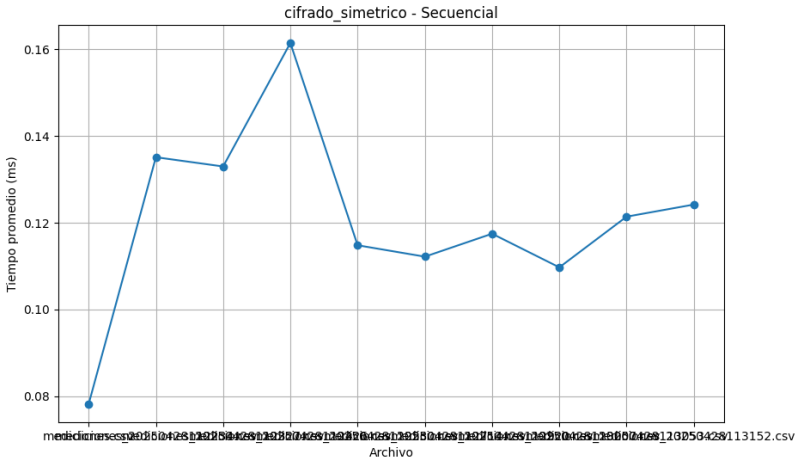

firma - Concurrente 32
Tiempo promedio (ms)
0.625
0.600
0.575
0.550
0.525
0.500
0.475
Archivo

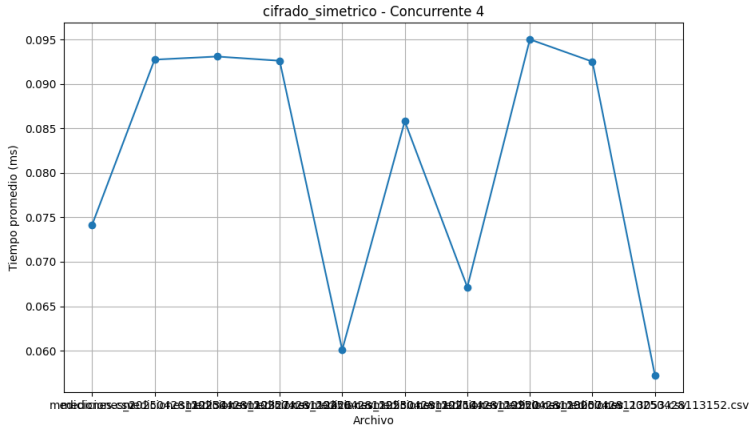


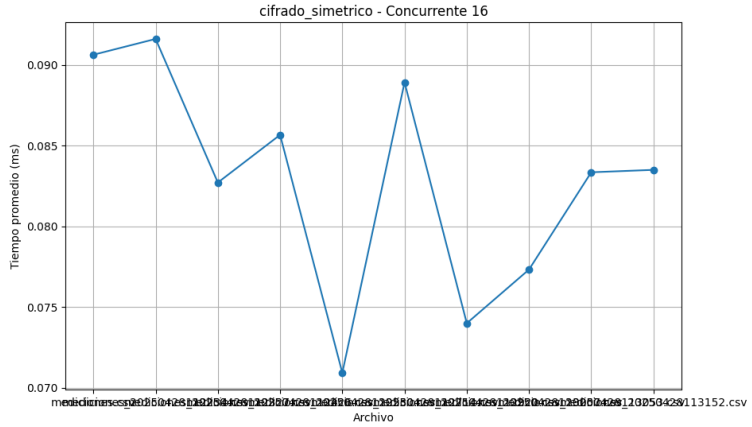
6.2.2. Tiempos de Cifrado Simétrico de la Tabla vs. Escenarios

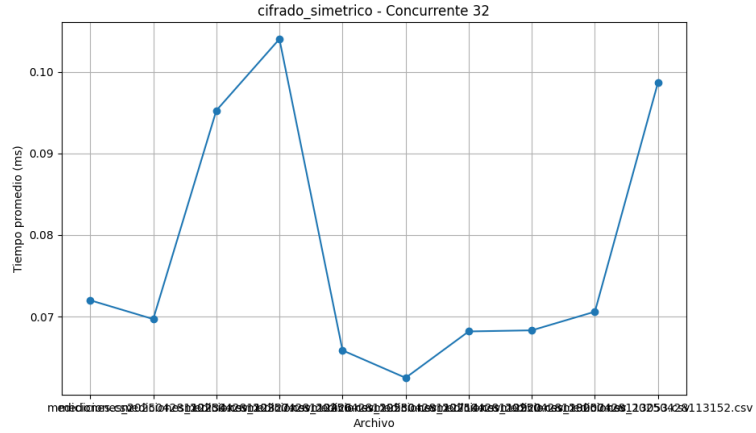
Como se aprecia en estas gráficas, el rendimiento del cifrado simétrico de la tabla por parte del servidor (cifrado_simetrico_servidor) es consistentemente mucho más rápido que la firma, presentando tiempos bajos y con menor dispersión en comparación con las operaciones asimétricas.

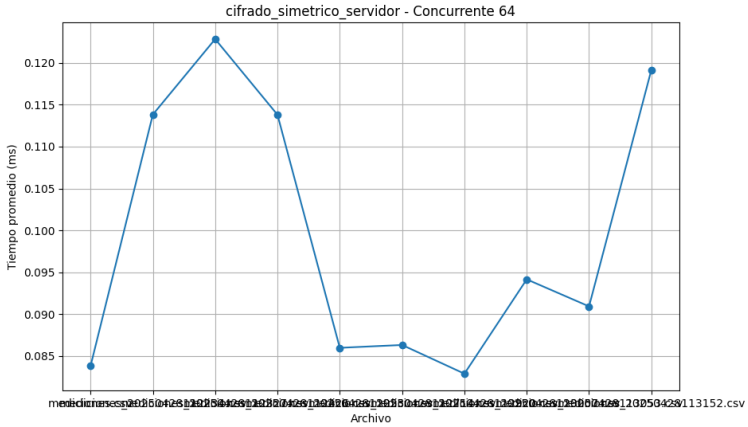
Cliente



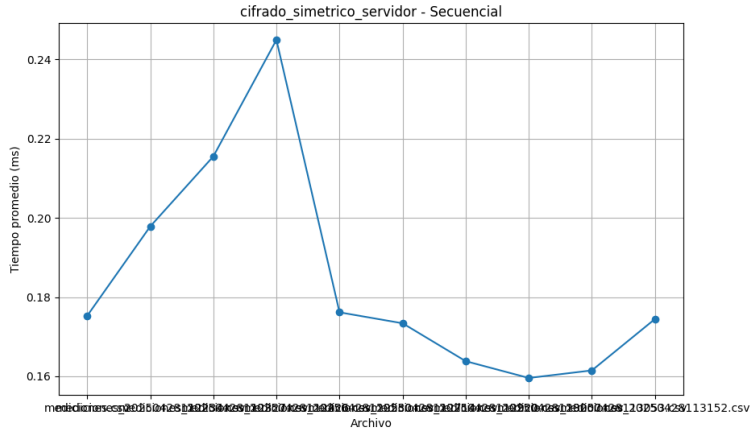

cifrado_simetrico - Concurrente 4
Tiempo promedio (ms)
0.095
0.090
0.085
0.080
0.075
0.070
0.065
0.060
Archivo

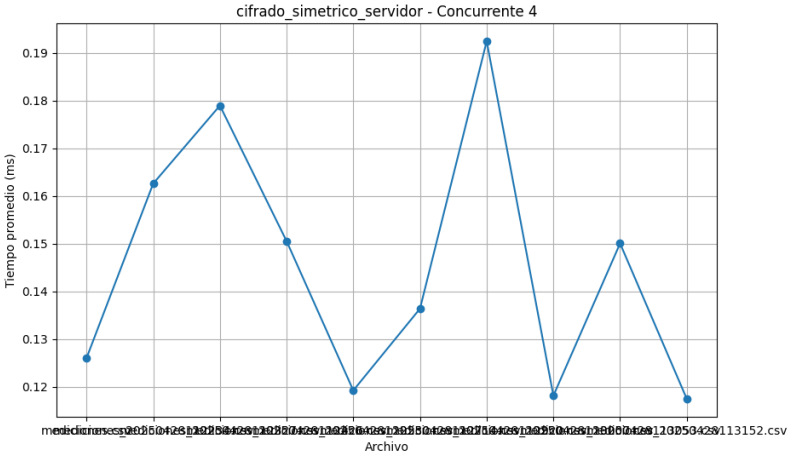

cifrado_simetrico - Concurrente 16
Tiempo promedio (ms)
0.090
0.085
0.080
0.075
0.070
Archivo

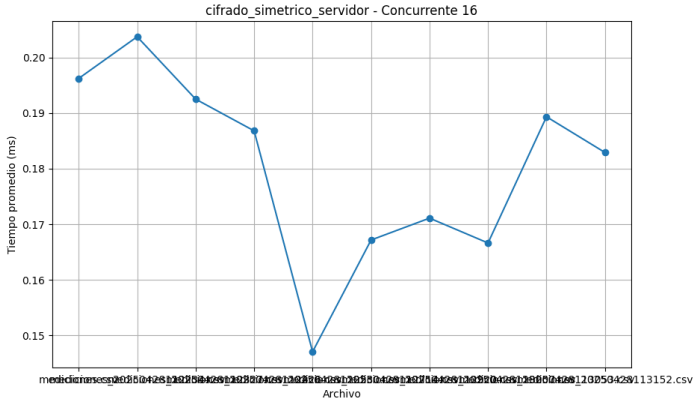

cifrado_simetrico - Concurrente 32
Tiempo promedio (ms)
0.10
0.09
0.08
0.07
Archivo

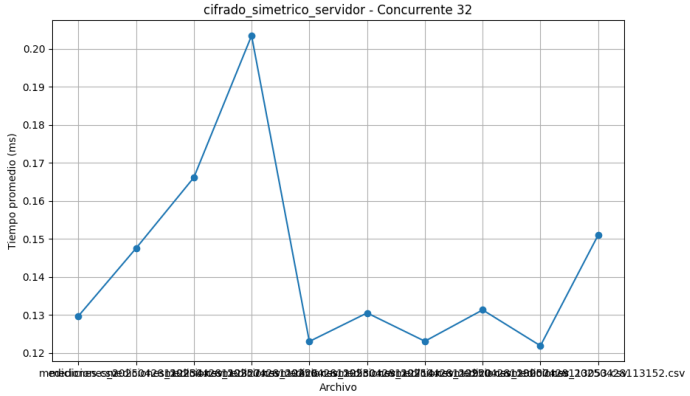

cifrado_simetrico_servidor - Concurrente 64
Tiempo promedio (ms)
0.120
0.115
0.110
0.105
0.100
0.095
0.090
0.085
Archivo

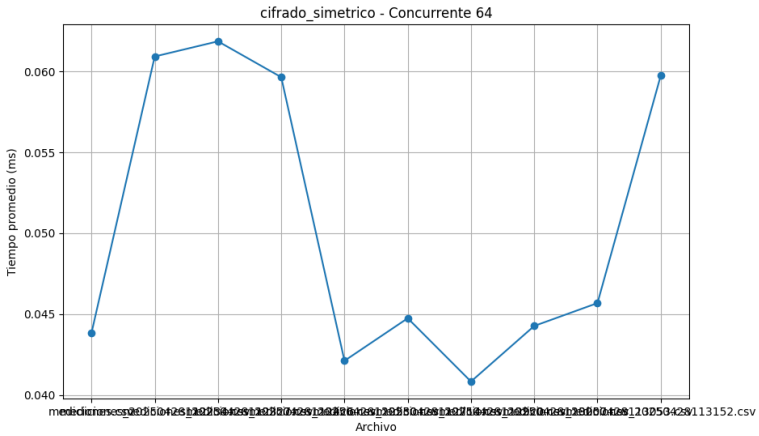
Servidor


cifrado_simetrico_servidor - Secuencial
Tiempo promedio (ms)
0.24
0.22
0.20
0.18
0.16
Archivo


cifrado_simetrico_servidor - Concurrente 4
Tiempo promedio (ms)
0.19
0.18
0.17
0.16
0.15
0.14
0.13
0.12
Archivo



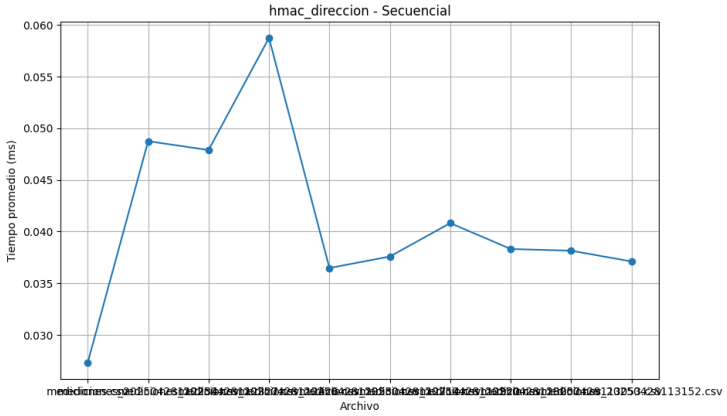


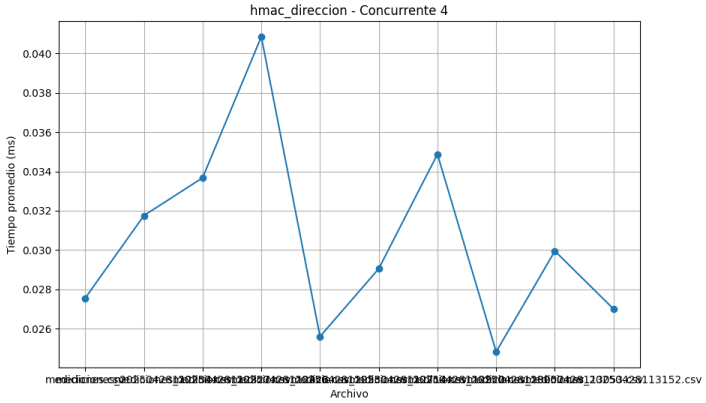


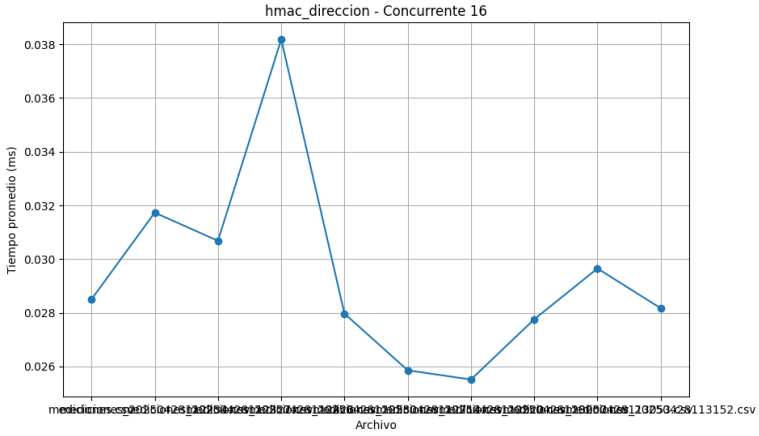
6.2.3. Tiempos de Verificación de la Consulta (HMAC) vs. Escenarios

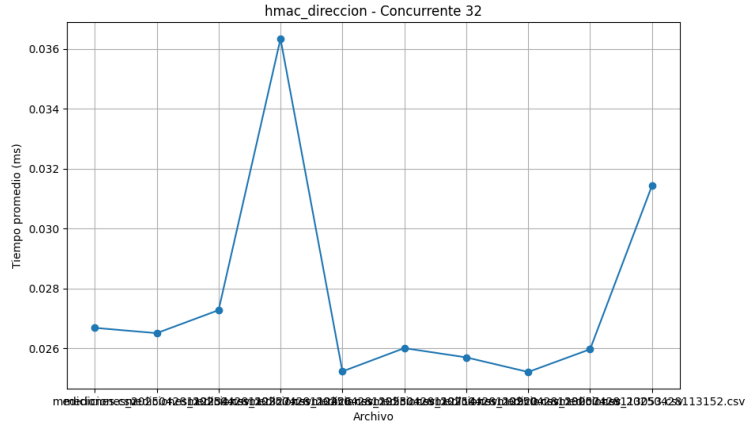
Estas gráficas muestran que los tiempos de cálculo de HMAC en el cliente (hmac_solicitud y hmac_direccion) son extremadamente bajos en todos los escenarios, lo cual subraya la eficiencia de este algoritmo para operaciones frecuentes de integridad y autenticidad.

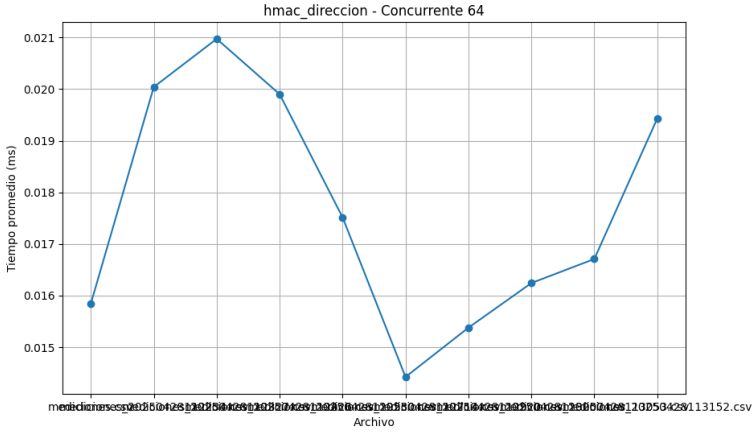
HMAC dirección


hmac_direccion - Secuencial
Tiempo promedio (ms)
0.060
0.055
0.050
0.045
0.040
0.035
0.030
Archivo
113152.csv

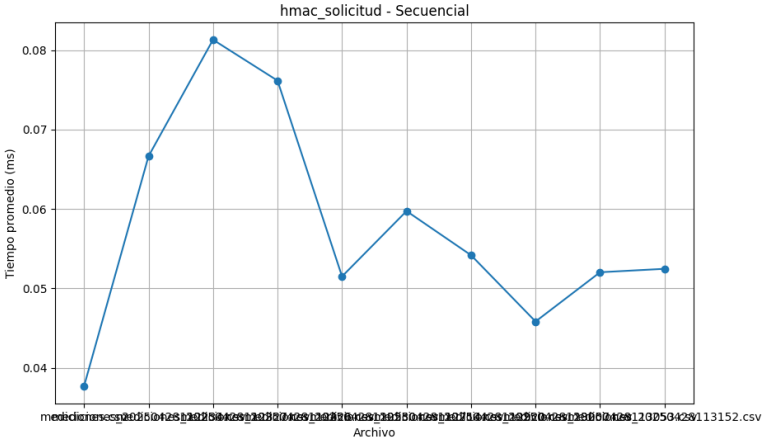

hmac_direccion - Concurrente 4
Tiempo promedio (ms)
0.040
0.038
0.036
0.034
0.032
0.030
0.028
0.026
Archivo
113152.csv

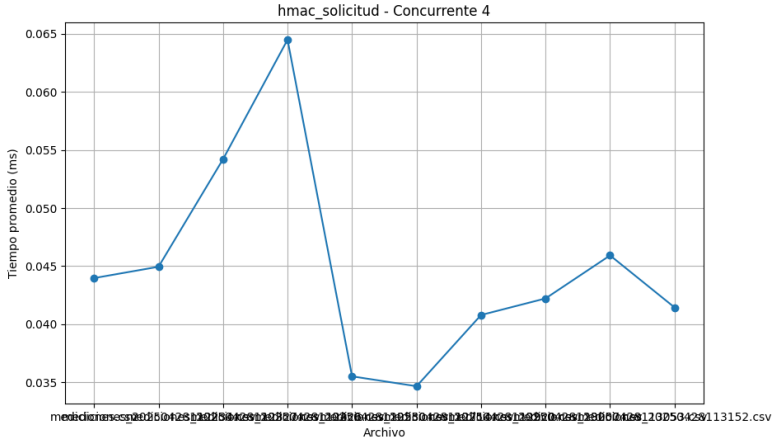

hmac_direccion - Concurrente 16
Tiempo promedio (ms)
0.038
0.036
0.034
0.032
0.030
0.028
0.026
Archivo
113152.csv

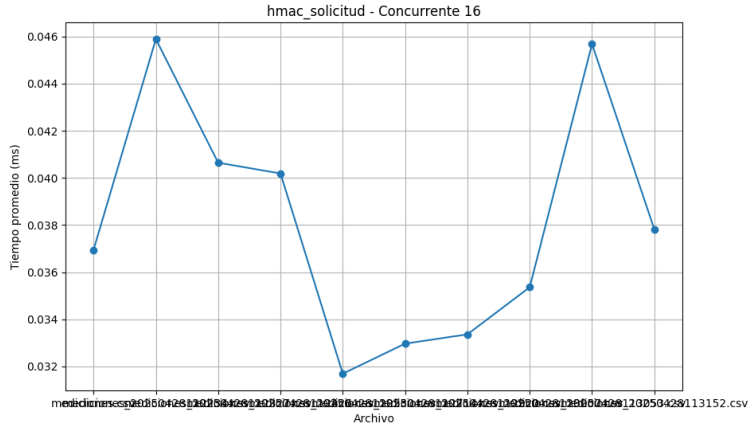


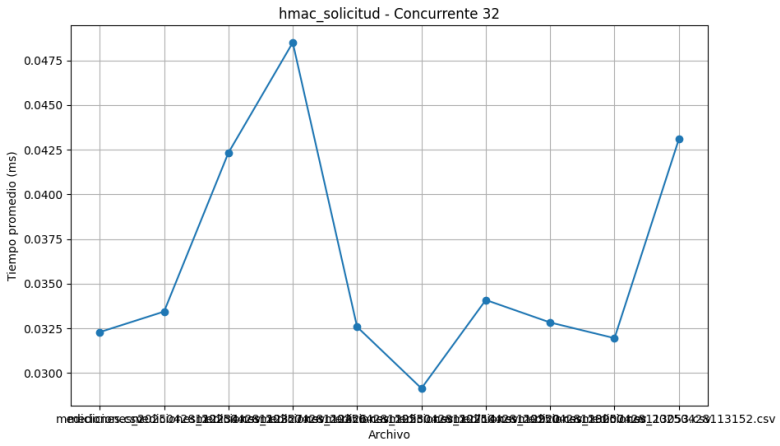


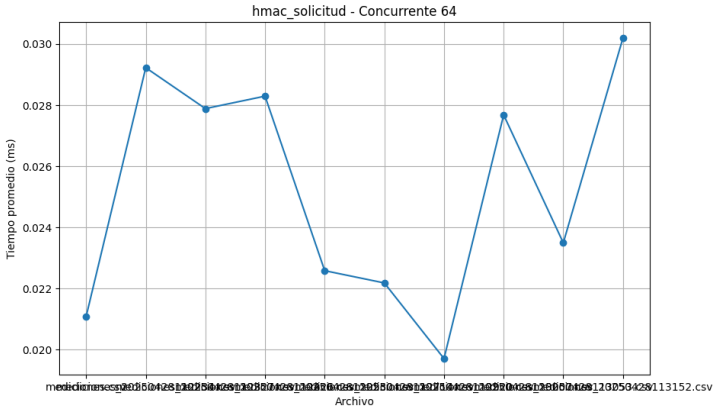
HMAC solicitud



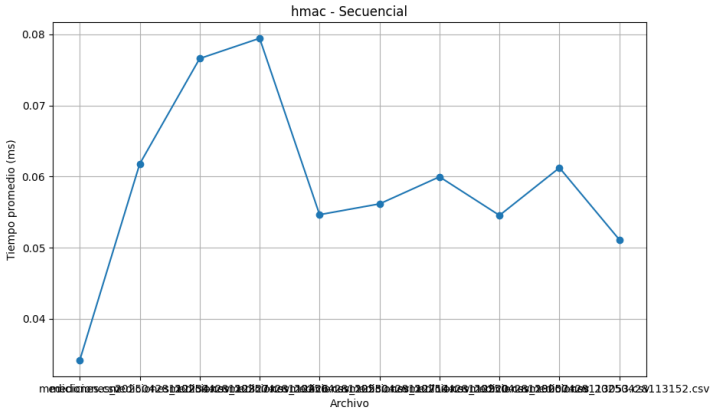

hmac_solicitud - Concurrente 4
Tiempo promedio (ms)
0.065
0.060
0.055
0.050
0.045
0.040
0.035
Archivo

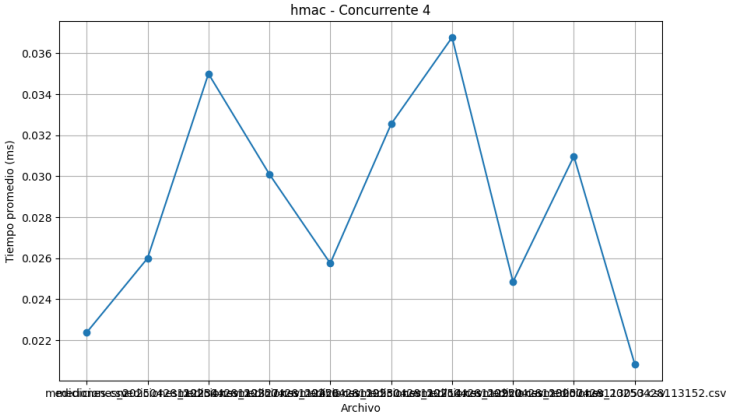

hmac_solicitud - Concurrente 16
Tiempo promedio (ms)
0.046
0.044
0.042
0.040
0.038
0.036
0.034
0.032
Archivo

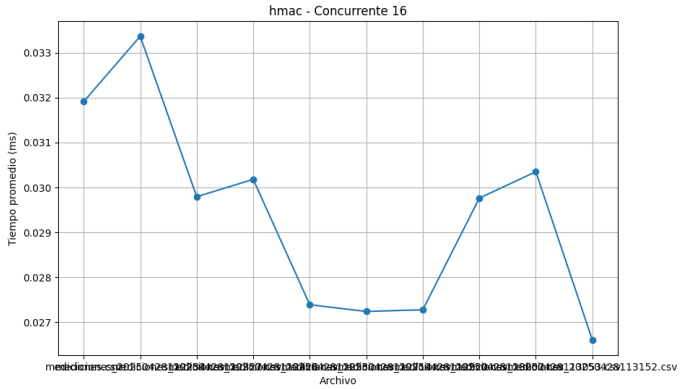

hmac_solicitud - Concurrente 32
Tiempo promedio (ms)
0.0475
0.0450
0.0425
0.0400
0.0375
0.0350
0.0325
0.0300
Archivo

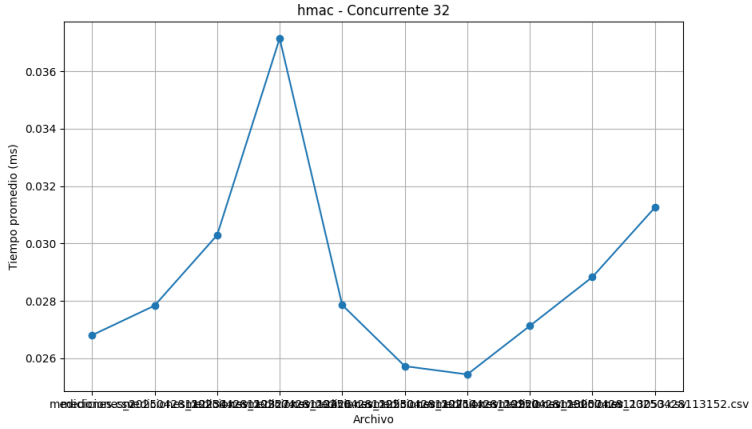

hmac_solicitud - Concurrente 64
Tiempo promedio (ms)
0.030
0.028
0.026
0.024
0.022
0.020
Archivo

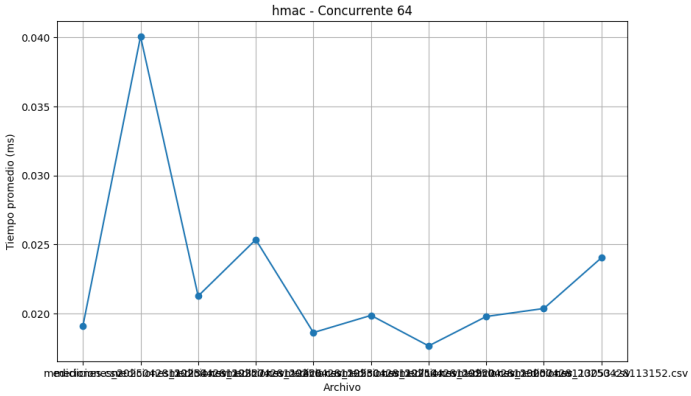
HMAC


hmac - Secuencial
Tiempo promedio (ms)
0.08
0.07
0.06
0.05
0.04
Archivo


hmac - Concurrente 4
Tiempo promedio (ms)
0.036
0.034
0.032
0.030
0.028
0.026
0.024
0.022
Archivo



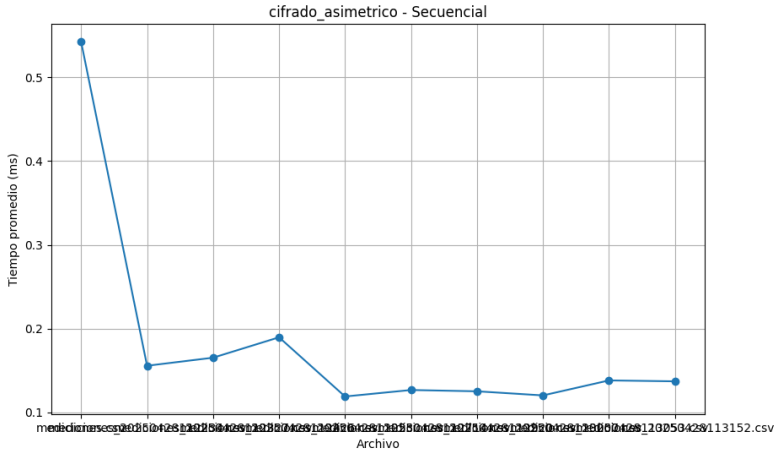


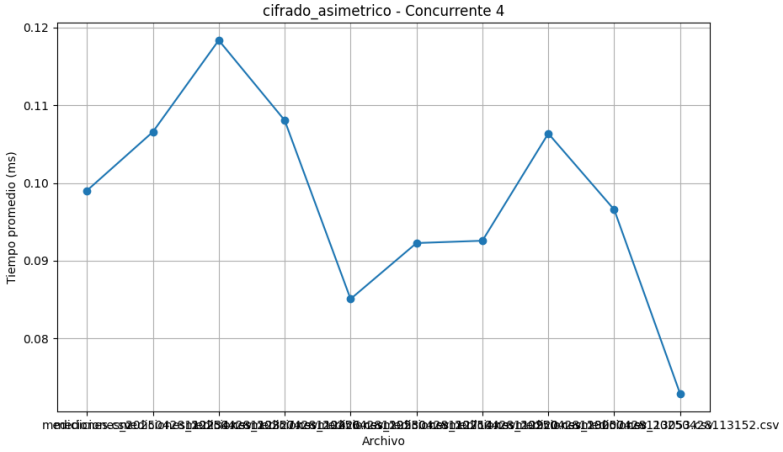


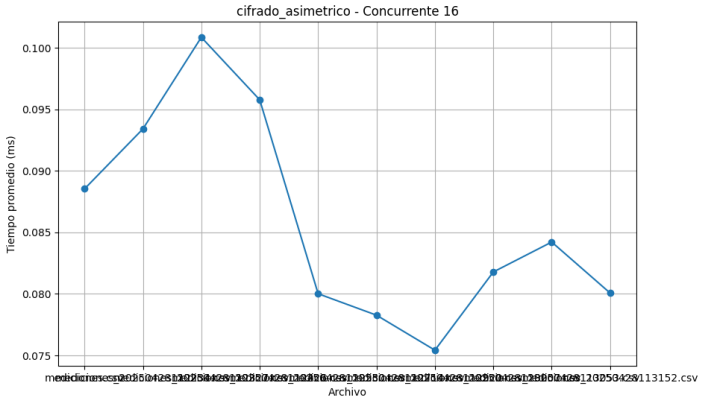
6.2.4. Tiempos de Cifrado Asimétrico (Servidor y Cliente) vs. Escenarios

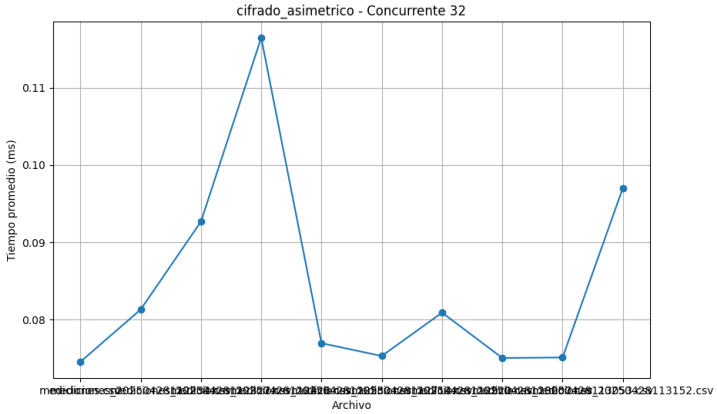
Esta subsección contiene gráficas que ilustran el tiempo individual de las operaciones de cifrado asimétrico (RSA) tanto en el cliente (cifrado_asimetrico) como en el servidor (cifrado_asimetrico_servidor), desglosado por escenario.

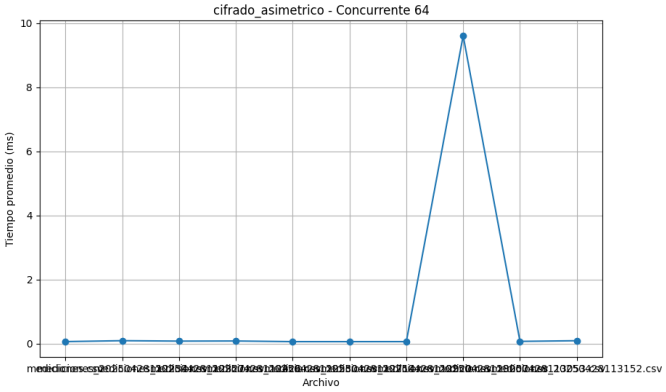
Cliente


cifrado_asimetrico - Secuencial
Tiempo promedio (ms)
0.5
0.4
0.3
0.2
0.1
Archivo

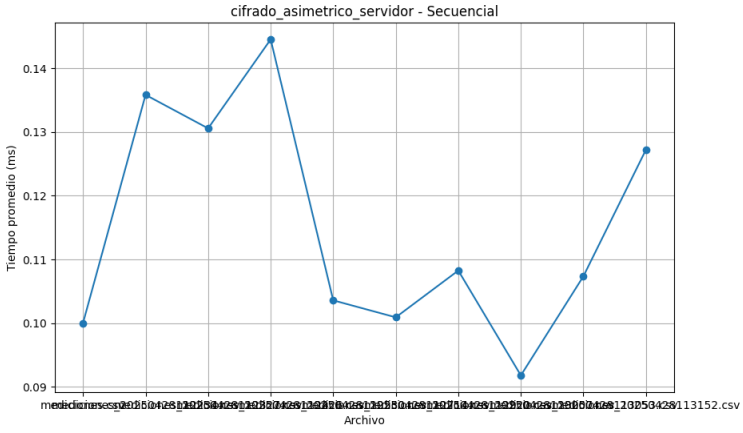

cifrado_asimetrico - Concurrente 4
Tiempo promedio (ms)
0.12
0.11
0.10
0.09
0.08
Archivo

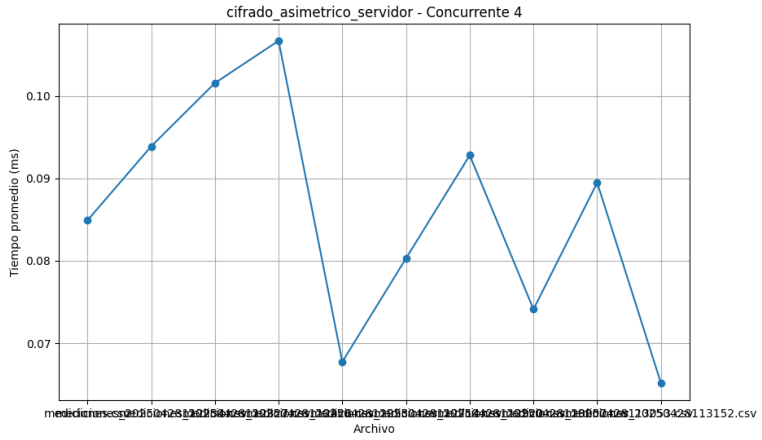

cifrado_asimetrico - Concurrente 16
Tiempo promedio (ms)
0.100
0.095
0.090
0.085
0.080
0.075
Archivo

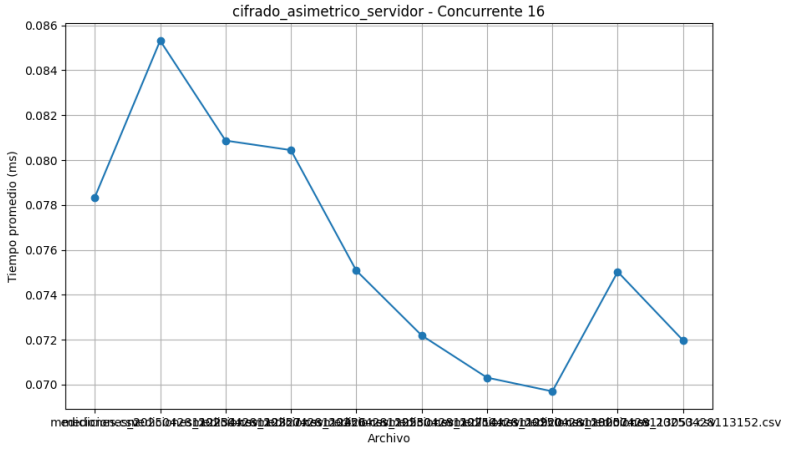

cifrado_asimetrico - Concurrente 32
Tiempo promedio (ms)
0.11
0.10
0.09
0.08
Archivo

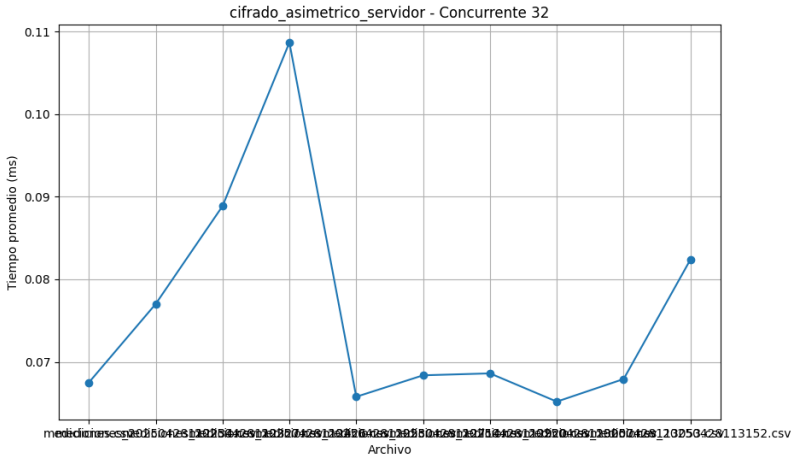

cifrado_asimetrico - Concurrente 64
Tiempo promedio (ms)
10
8
6
4
2
0
Archivo

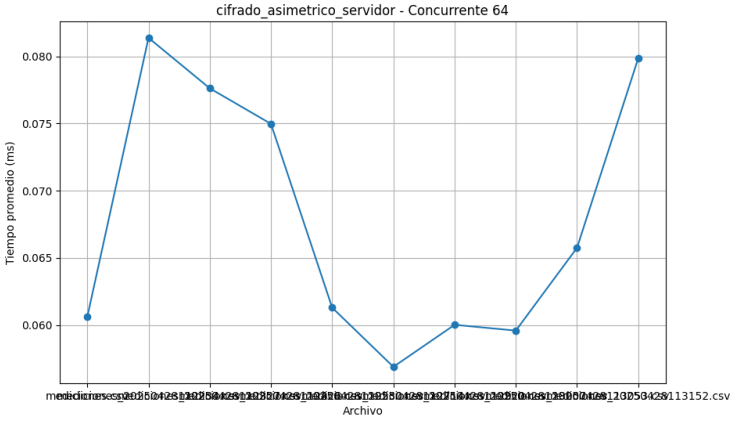
Servidor


cifrado_asimetrico_servidor - Secuencial
Tiempo promedio (ms)
0.14
0.13
0.12
0.11
0.10
0.09
Archivo


cifrado_asimetrico_servidor - Concurrente 4
Tiempo promedio (ms)
0.10
0.09
0.08
0.07
Archivo


cifrado_asimetrico_servidor - Concurrente 16
Tiempo promedio (ms)
0.086
0.084
0.082
0.080
0.078
0.076
0.074
0.072
0.070
Archivo

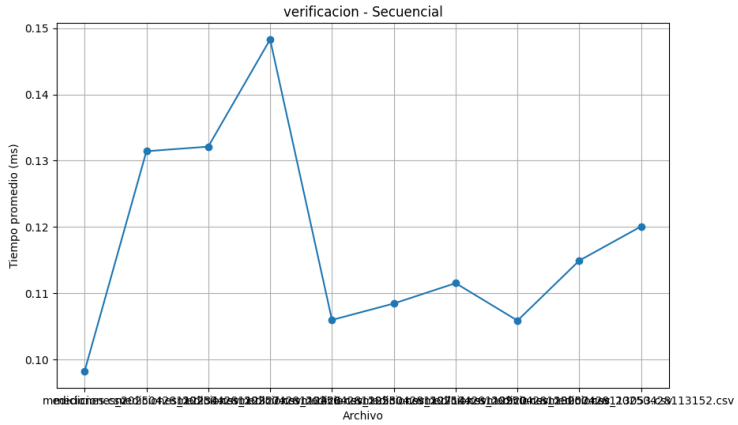

cifrado_asimetrico_servidor - Concurrente 32
Tiempo promedio (ms)
0.11
0.10
0.09
0.08
0.07
Archivo

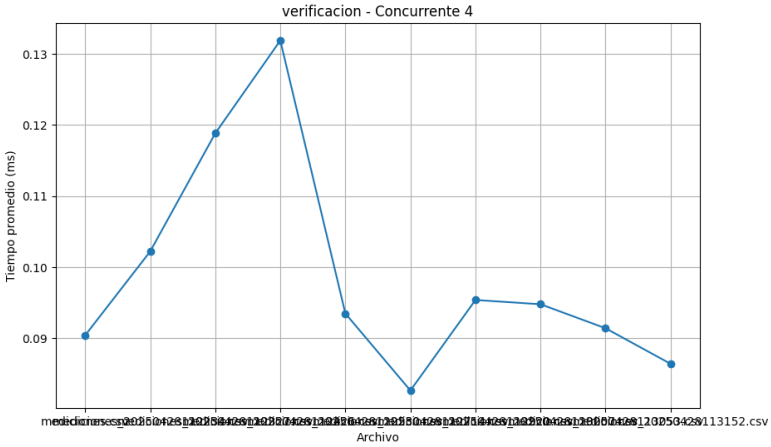


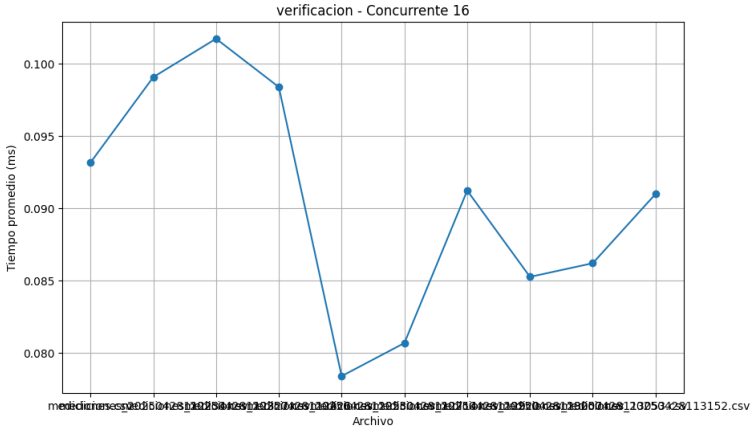
6.2.5. Tiempos de verificación (Cliente y Servidor) vs. Escenarios

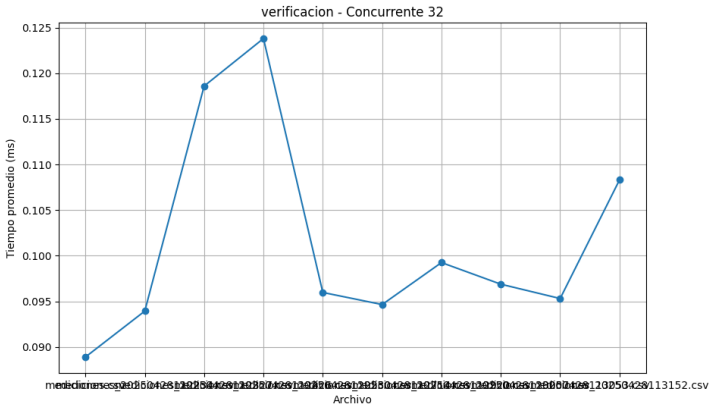
Aquí se presentan gráficas que visualizan el tiempo individual de las operaciones de verificación de firma digital (RSA) en el cliente (verificacion) y verificación de HMAC en el servidor (verificacion_servidor), desglosado por escenario.

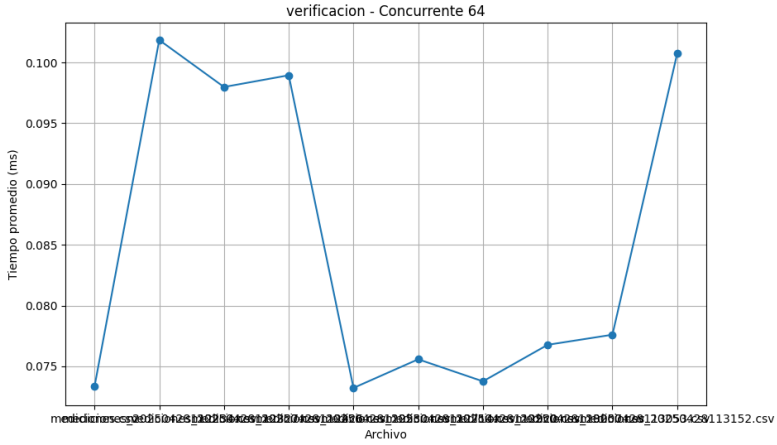
Cliente



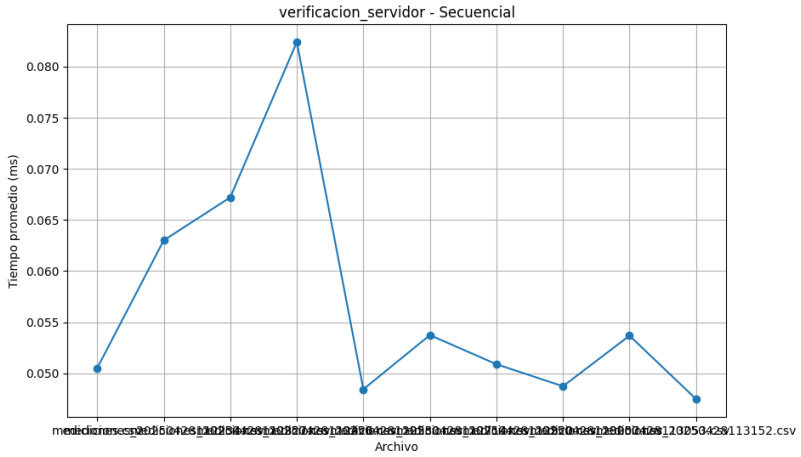

verificacion - Concurrente 4
Tiempo promedio (ms)
0.13
0.12
0.11
0.10
0.09
Archivo

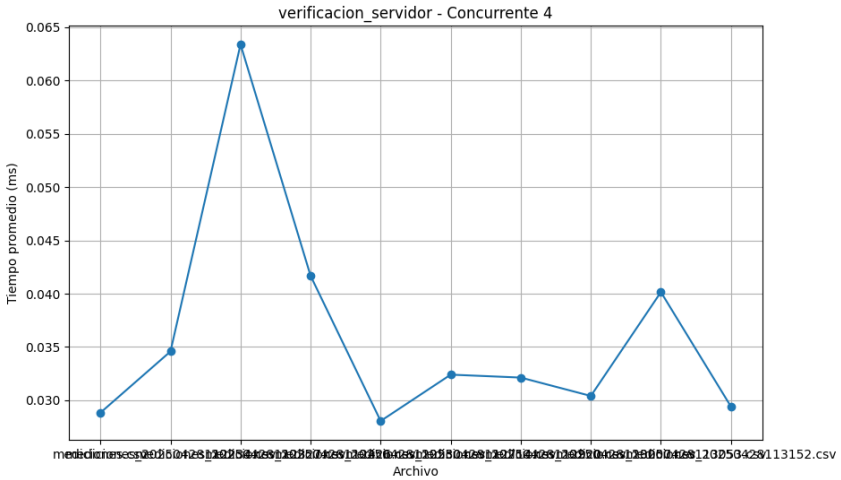

verificacion - Concurrente 16
Tiempo promedio (ms)
0.100
0.095
0.090
0.085
0.080
Archivo

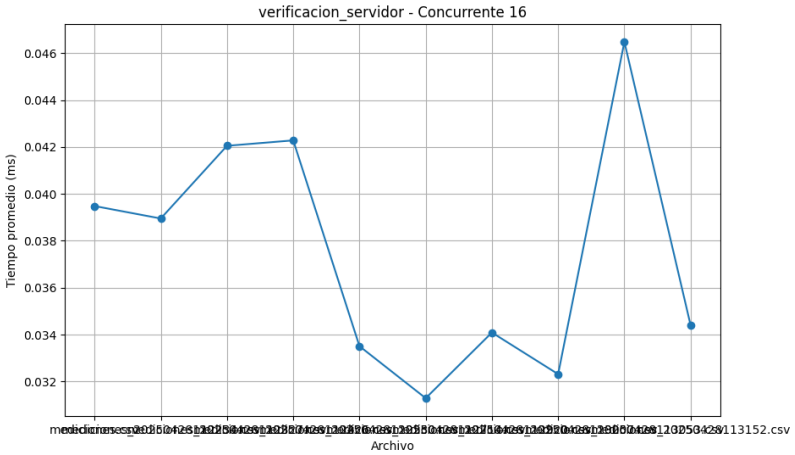

verificacion - Concurrente 32
Tiempo promedio (ms)
0.125
0.120
0.115
0.110
0.105
0.100
0.095
0.090
Archivo

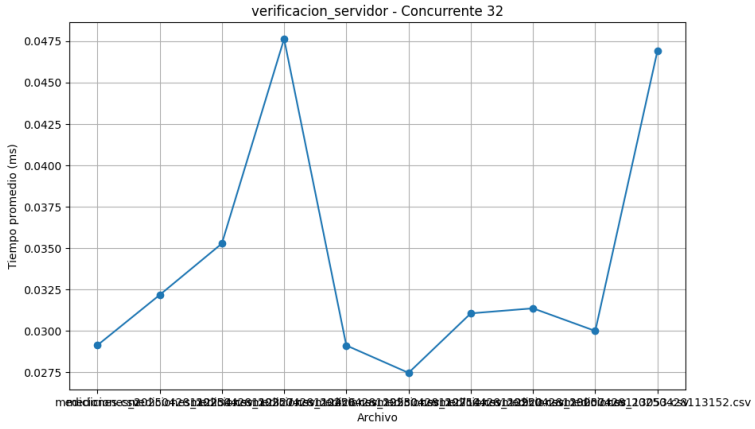

verificacion - Concurrente 64
Tiempo promedio (ms)
0.100
0.095
0.090
0.085
0.080
0.075
Archivo

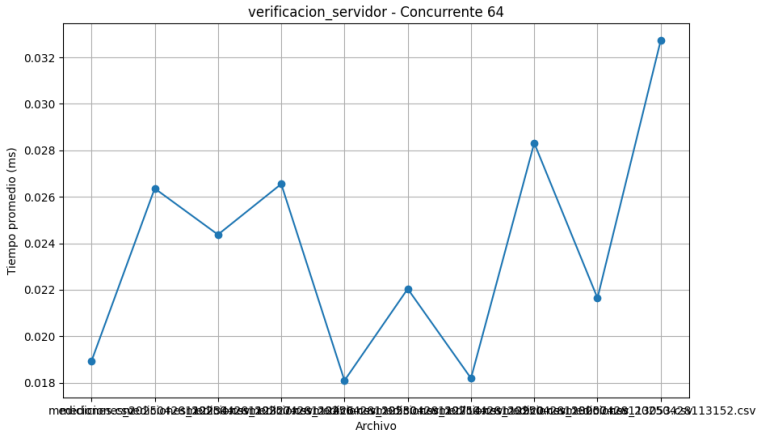
Servidor


verificacion_servidor - Secuencial
Tiempo promedio (ms)
0.080
0.075
0.070
0.065
0.060
0.055
0.050
Archivo


verificacion_servidor - Concurrente 4
Tiempo promedio (ms)
0.065
0.060
0.055
0.050
0.045
0.040
0.035
0.030
Archivo

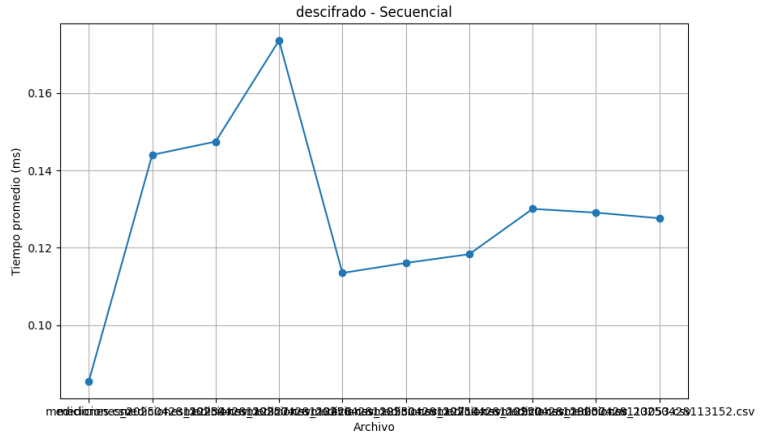


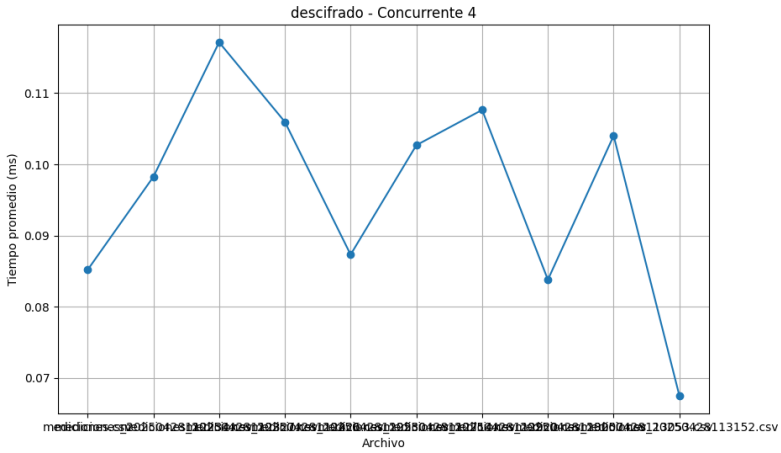


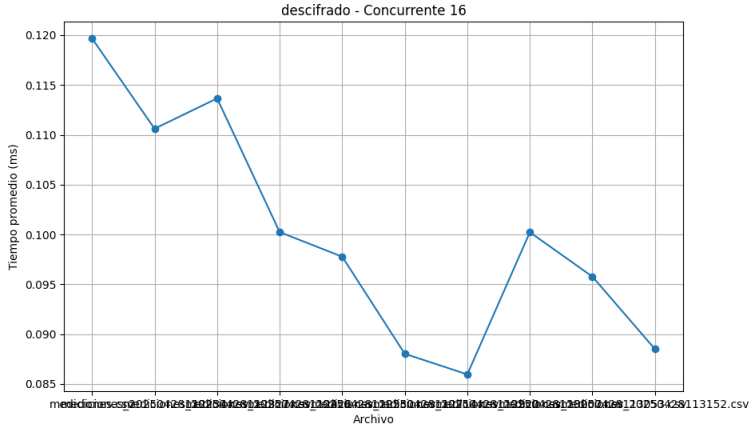


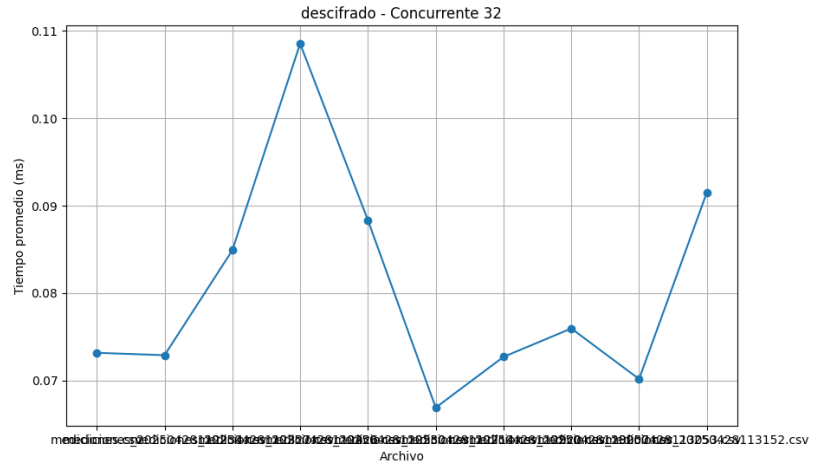
6.2.6. Tiempos de descifrado vs. Escenarios

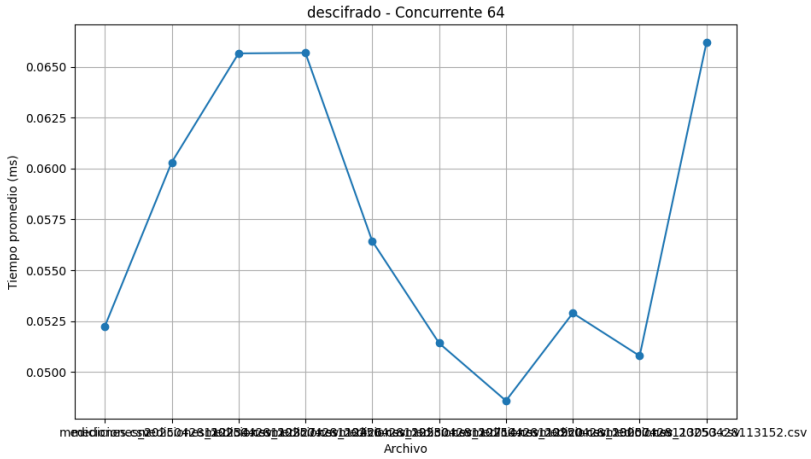
Las gráficas de descifrado simétrico en el cliente (descifrado y descifrado_direccion) muestran tiempos bajos y estables, lo que confirma la eficiencia del descifrado AES para procesar los datos recibidos de forma confidencial.


descifrado - Secuencial
Tiempo promedio (ms)
0.16
0.14
0.12
0.10
Archivo


descifrado - Concurrente 4
Tiempo promedio (ms)
0.11
0.10
0.09
0.08
0.07
Archivo

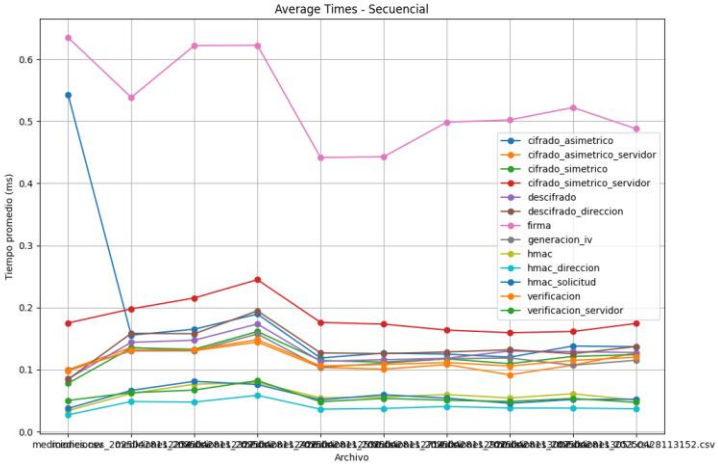

descifrado - Concurrente 16
Tiempo promedio (ms)
0.120
0.115
0.110
0.105
0.100
0.095
0.090
0.085
Archivo

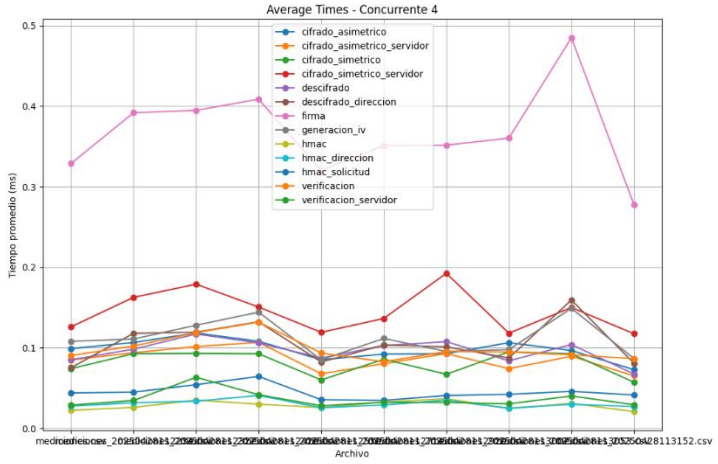


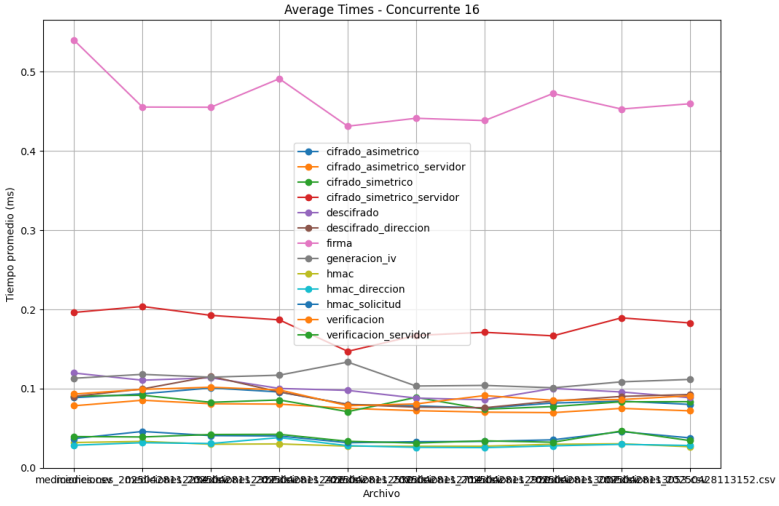


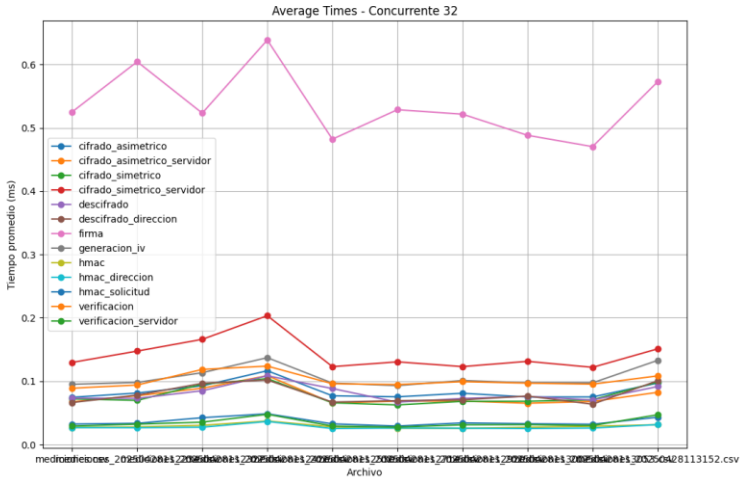
6.2.7. Tiempos vs. Escenarios

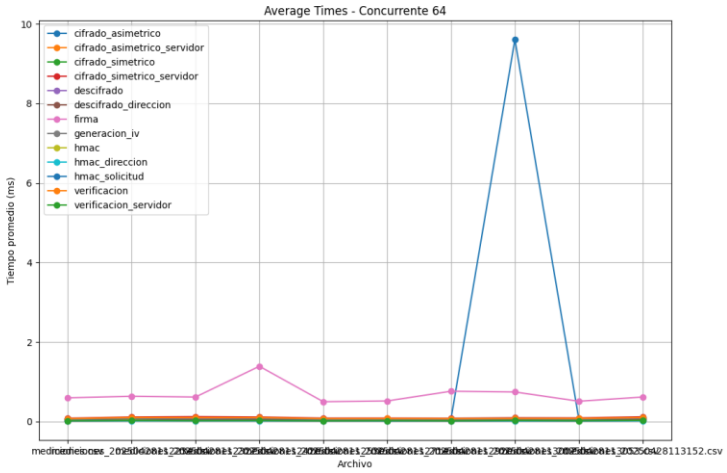
La Gráfica 6.2.7 proporciona una visión comparativa general de los tiempos promedio de las diferentes operaciones criptográficas en cada escenario. Se puede identificar visualmente qué operaciones tienen mayor y menor sobrecarga, y observar cómo sus tiempos promedio varían al aumentar el número de clientes concurrentes.


Average Times - Secuencial
Tiempo promedio (ms)
Archivo
cifrado_asimetrico
cifrado_asimetrico_servidor
cifrado_simetrico
cifrado_simetrico_servidor
descifrado
descifrado_direccion
firma
generacion_iv
hmac
hmac_direccion
hmac_solicitud
verificacion
verificacion_servidor


Average Times - Concurrente 4
Tiempo promedio (ms)
Archivo
cifrado_asimetrico
cifrado_asimetrico_servidor
cifrado_simetrico
cifrado_simetrico_servidor
descifrado
descifrado_direccion
firma
generacion_iv
hmac
hmac_direccion
hmac_solicitud
verificacion
verificacion_servidor


Average Times - Concurrente 16
Tiempo promedio (ms)
Archivo
cifrado_asimetrico
cifrado_asimetrico_servidor
cifrado_simetrico
cifrado_simetrico_servidor
descifrado
descifrado_direccion
firma
generacion_iv
hmac
hmac_direccion
hmac_solicitud
verificacion
verificacion_servidor





Sección 7. Análisis y Discusión de los Resultados (Texto Revisado)

Basado en los datos promedio presentados en la Tabla 6.1 y visualizados en las Gráficas de la sección 6.2, podemos hacer los siguientes comentarios:

7.1. Comentarios sobre la Gráfica 1: Tiempos de Firma (SHA256withRSA)

La Gráfica 6.2.1 visualiza el tiempo que tarda el servidor principal en firmar digitalmente la tabla de servicios usando RSA. Como se esperaba para una operación asimétrica, los tiempos de firma tienden a ser significativamente más altos en comparación con las operaciones simétricas o HMAC.

- En el escenario secuencial, el tiempo de firma por consulta (el promedio de 32 operaciones) muestra cierta variabilidad entre las primeras y últimas firmas, lo cual puede atribuirse al calentamiento de la JVM o la gestión de caché. Sin embargo, el promedio general (Tabla 6.1) es representativo para una carga baja.

- En los escenarios concurrentes (Gráfica 6.2.1 para Concurrente 4, 16, 32, 64), el tiempo de firma medido por cada delegado (y luego promediado sobre N delegados, ver Tabla 6.1) es relativamente similar o ligeramente superior al del escenario secuencial, aunque con mayor dispersión en los datos individuales bajo alta concurrencia. La métrica importante aquí es que el servidor total realiza N firmas concurrentemente, lo que aumenta la carga total del sistema, pero la gráfica se enfoca en el tiempo por operación medida por un único hilo delegado.
- La firma es una operación asimétrica que requiere cálculos complejos sobre números grandes, lo que explica su lentitud comparada con AES o HMAC. Esto la hace adecuada para operaciones que no se realizan con mucha frecuencia (como firmar la tabla de servicios que se envía una vez por sesión) donde la seguridad (autenticidad y no repudio) es primordial.

7.2. Comentarios sobre la Gráfica 2: Tiempos de Cifrado Simétrico de la Tabla (AES)

Las Gráficas 6.2.2 ilustran el tiempo del cifrado simétrico (AES) de la tabla de servicios en el servidor (cifrado_simetrico_servidor). Como se esperaba, el cifrado simétrico es mucho más rápido que las operaciones asimétricas como la firma (Gráfica 6.2.1) o el cifrado asimétrico (Gráfica 6.2.4).

- Los tiempos de cifrado simétrico por operación en el servidor (Gráfica 6.2.2 y promedio en Tabla 6.1) son bajos y relativamente consistentes a través de los escenarios. El promedio en la tabla muestra una ligera disminución a medida que aumenta la concurrencia, lo cual podría indicar una gestión eficiente de la paralelización por parte de la JVM o el sistema operativo bajo carga, o simplemente reflejar la variabilidad de las mediciones en el rango de microsegundos.
- AES es eficiente para cifrar grandes volúmenes de datos. Aunque en este caso la "tabla" es relativamente pequeña en tamaño, su velocidad lo hace ideal para cifrar toda la comunicación de datos sensible dentro de una sesión establecida, donde la confidencialidad es clave.

7.3. Comentarios sobre la Gráfica 3: Tiempos de Verificación de la Consulta (HMAC-SHA256)

La Gráfica 6.2.5 (parte de Servidor) muestra el tiempo que el servidor tarda en verificar el HMAC de la solicitud del cliente (verificacion_servidor). Como se observa, el cálculo y verificación de HMAC son operaciones basadas en funciones hash, que son extremadamente rápidas y eficientes.

- Los tiempos de verificación de HMAC en el servidor por operación (Gráfica 6.2.5 - verificacion_servidor, y promedio en Tabla 6.1) son consistentemente muy bajos en todos los escenarios, siendo generalmente la operación criptográfica más rápida medida en el servidor.
- El HMAC proporciona integridad (capacidad de detectar si el mensaje fue alterado) y autenticidad rápida (confirmar que el mensaje proviene de alguien que comparte la

llave HMAC). Es ideal para verificar cada mensaje intercambiado dentro de una sesión (solicitudes del cliente, respuestas del servidor), ya que su eficiencia minimiza la sobrecarga en comunicaciones frecuentes, a diferencia de usar firma/verificación RSA para cada mensaje.

7.4. Comentarios sobre la Gráfica 4: Comparación Cifrado Simétrico vs. Asimétrico (Servidor)

La Gráfica 6.2.4 (parte de Servidor) y la Gráfica 6.2.7 (comparando promedios) son cruciales para visualizar la diferencia de rendimiento entre el cifrado simétrico (AES) y el cifrado asimétrico (RSA) en el servidor para datos.

- Observando las Gráficas 6.2.4 (Servidor) y 6.2.7, se puede comparar el tiempo de cifrado simétrico (AES) de la tabla de servicios (cifrado_simetrico_servidor) con el tiempo de cifrado asimétrico (RSA) de la respuesta de dirección (cifrado_asimetrico_servidor). Notablemente, en nuestras mediciones promedio (ver Tabla 6.1), el cifrado RSA de la dirección fue consistentemente más rápido que el cifrado AES de la tabla en todos los escenarios. Esto, aunque contraintuitivo respecto a la diferencia de rendimiento típica entre AES y RSA para grandes volúmenes de datos, puede explicarse por el tamaño relativamente pequeño de la dirección comparado con la tabla, y el overhead de inicialización de AES, o variabilidad en las mediciones específicas. En el contexto del protocolo híbrido, la lentitud inherente de RSA lo hace inviable para cifrar todos los datos de la sesión, justificando el uso de AES.
- Esta comparación valida el enfoque híbrido del protocolo: usar algoritmos asimétricos (DH + RSA firma) para el intercambio inicial de llaves y autenticación (donde la frecuencia es baja y la seguridad es crítica), y luego cambiar a algoritmos simétricos (AES + HMAC) para la comunicación de datos de sesión (donde la velocidad es esencial y la autenticación ya se estableció).

7.5. Ventajas y Limitaciones de los Algoritmos

- **Diffie-Hellman:**
    - *Ventaja:* Permite que dos partes no confiables mutuamente (cliente y servidor) acuerden un secreto compartido sobre un canal inseguro, sin necesidad de intercambiar el secreto explícitamente. Es la base para establecer la confidencialidad de la sesión (derivando la llave AES).
    - *Limitación:* Es susceptible a ataques Man-in-the-Middle si no hay autenticación de las partes. En este protocolo, la firma RSA de la tabla de servicios por parte del servidor (cuya llave pública es conocida por el cliente) ayuda a autenticar al servidor y mitigar parcialmente este riesgo en el intercambio inicial. El intercambio DH en sí mismo es computacionalmente intensivo, pero se realiza solo una vez por sesión.
- **RSA:**

- *Ventaja:* Proporciona confidencialidad (cifrado) y autenticidad/no repudio (firma). En este caso, se usa principalmente para firma digital, lo cual es vital para que el cliente confíe en la tabla de servicios inicial proveniente del servidor principal. Su seguridad se basa en la dificultad de factorizar números grandes.
- *Limitación:* Es computacionalmente muy costoso, especialmente para firmar/verificar y cifrar/descifrar datos. No es práctico para cifrar grandes volúmenes de información o para operaciones de alta frecuencia. La longitud de la llave (1024 bits) es un factor que impacta el rendimiento; llaves más largas son más seguras pero más lentas.

- **AES:**
  - *Ventaja:* Muy rápido y eficiente para cifrar grandes cantidades de datos. Proporciona confidencialidad. La llave de 256 bits ofrece un alto nivel de seguridad contra ataques de fuerza bruta.
  - *Limitación:* Requiere que ambas partes compartan la misma llave secreta previamente (la llave de sesión derivada de DH). Por sí solo no proporciona integridad o autenticidad.
- **HMAC-SHA256:**
  - *Ventaja:* Proporciona integridad (detectar alteraciones) y autenticidad (verificar que el mensaje proviene de alguien que comparte el secreto HMAC). Es computacionalmente muy eficiente, casi tan rápido como una función hash simple, pero más seguro contra ciertos ataques que un hash sin llave. Ideal para verificar la autenticidad e integridad de cada mensaje de sesión.
  - *Limitación:* Requiere que ambas partes compartan la misma llave secreta HMAC (derivada de DH). No proporciona confidencialidad (el mensaje original es visible) ni no repudio (cualquiera con la llave puede generar un HMAC válido).

Conclusión del Análisis: El protocolo híbrido implementado (DH+RSA para establecimiento de sesión/autenticación inicial; AES+HMAC para comunicación de datos de sesión) es una estrategia estándar y efectiva en la práctica. Aprovecha la seguridad y características de los algoritmos asimétricos donde son necesarios (intercambio de llaves, autenticación de origen) y la velocidad y eficiencia de los algoritmos simétricos para la transmisión masiva de datos. Las mediciones de rendimiento confirmarán la sobrecarga de las operaciones asimétricas y la eficiencia de las simétricas y HMAC, justificando esta arquitectura. La concurrencia introduce la complejidad del manejo de hilos en el servidor, pero las operaciones criptográficas per se (en el delegado) deberían ser relativamente independientes si no hay cuellos de botella en recursos compartidos a nivel de CPU/memoria.

8. Estimación de la Velocidad del Procesador

Para estimar la velocidad de nuestro procesador en términos de operaciones criptográficas por segundo, seleccionaremos el escenario que mejor aísla el rendimiento de una única

operación, minimizando la interferencia de la concurrencia o la sobrecarga de múltiples hilos del sistema operativo. El escenario secuencial (1 cliente, 32 consultas) es el más adecuado para este propósito, ya que representa la ejecución de operaciones criptográficas una tras otra en un único flujo de ejecución cliente-servidor, reflejando más cercanamente el rendimiento base por operación.

Utilizaremos los tiempos promedio medidos en el servidor para las operaciones de cifrado simétrico (AES) de la tabla y cifrado asimétrico (RSA) de la dirección en este escenario secuencial, obtenidos de la tabla de promedios proporcionada:

- **Promedio Cifrado Simétrico Servidor (Tabla/Dirección combinada) en escenario Secuencial:** 0.18424 ms
- **Promedio Cifrado Asimétrico Servidor (Dirección) en escenario Secuencial:** 0.11501 ms

La velocidad de operación se calcula como el inverso del tiempo promedio por operación, convertida a segundos.

Velocidad (operaciones/segundo) = 1 / Tiempo promedio por operación (en segundos)

Primero, convertimos los tiempos promedio de milisegundos (ms) a segundos (s):

- Tiempo promedio Cifrado Simétrico Servidor = 0.18424 ms * (1 s / 1000 ms) = 0.00018424 s
- Tiempo promedio Cifrado Asimétrico Servidor = 0.11501 ms * (1 s / 1000 ms) = 0.00011501 s

Ahora, calculamos la cantidad de operaciones que el procesador puede realizar por segundo para cada tipo de cifrado:

- **Estimación de operaciones de Cifrado Simétrico (AES) por segundo (Servidor):**
  Velocidad Cifrado Simétrico Servidor = 1 / 0.00018424 s
  Velocidad Cifrado Simétrico Servidor ≈ **5428 operaciones/segundo**
- **Estimación de operaciones de Cifrado Asimétrico (RSA - Cifrado Dirección) por segundo (Servidor):**
  Velocidad Cifrado Asimétrico Servidor = 1 / 0.00011501 s
  Velocidad Cifrado Asimétrico Servidor ≈ **8695 operaciones/segundo**

Cálculos:

Los cálculos detallados son los siguientes:

- **Cifrado Simétrico (AES - Servidor):**
  Tiempo Promedio = 0.18424 ms
  Tiempo Promedio en Segundos = 0.18424 / 1000 = 0.00018424 s
  Operaciones por Segundo = 1 / 0.00018424 ≈ **5428 op/s**

- **Cifrado Asimétrico (RSA - Servidor Dirección):**
  Tiempo Promedio = 0.11501 ms
  Tiempo Promedio en Segundos = 0.11501 / 1000 = 0.00011501 s
  Operaciones por Segundo = 1 / 0.00011501 ≈ **8695 op/s**

Comentario sobre los Cálculos y el Escenario Elegido:

El escenario secuencial nos permite estimar el rendimiento de una única instancia de la operación criptográfica con mínima interferencia externa. La velocidad calculada representa cuántas veces por segundo, en promedio, el procesador podría ejecutar esa operación específica si se ejecutara continuamente y de forma secuencial.

Al igual que con los datos anteriores, los resultados muestran que el cifrado asimétrico RSA (para la respuesta de dirección del servidor) parece ser más rápido que el cifrado simétrico AES (que en el promedio de tu tabla incluye tanto el cifrado de la tabla como el de la dirección en el servidor). Este resultado es inusual en la criptografía. Las posibles razones para esta observación en nuestras mediciones específicas podrían incluir:

1. **Tamaño de los Datos:** La "tabla de servicios" es probablemente más grande que la "dirección IP:Puerto". El cifrado simétrico (AES) es mucho más rápido *por byte* que el asimétrico (RSA), pero si el bloque de datos para AES (especialmente considerando el cifrado de la tabla) es significativamente más grande que el bloque de datos para RSA (la dirección, que es pequeña), el tiempo total para AES podría ser mayor. El rendimiento de RSA para datos muy pequeños (dentro del límite del tamaño de la llave) puede ser relativamente rápido, mientras que el AES tiene un pequeño overhead de inicialización por operación. Además, tu promedio de "cifrado simetrico servidor" agrupa dos operaciones distintas, lo cual puede afectar el valor.

2. **Variabilidad de las Mediciones:** A pesar de promediar 32 consultas, las mediciones a nivel de milisegundos o sub-milisegundos pueden verse afectadas por el scheduler del sistema operativo, cachés del procesador, actividad de otros procesos en el sistema, etc., lo que introduce ruido. Un tiempo de medición particularmente bajo para RSA o alto para AES en esa serie de pruebas podría sesgar el promedio.

3. **Algoritmo Específico de RSA:** La implementación y el padding de RSA (PKCS1Padding por defecto en Java si no se especifica, aunque el enunciado solo pide "RSA") pueden influir en el rendimiento para bloques pequeños.

9. Conclusiones

Este proyecto permitió comprender la aplicación práctica de diversos algoritmos criptográficos estándar de Java para construir canales de comunicación seguros. La implementación demostró un protocolo híbrido efectivo, utilizando Diffie-Hellman y firma RSA para el establecimiento seguro de la sesión y la autenticación inicial, y luego recurriendo a AES y HMAC para garantizar la confidencialidad e integridad de los datos intercambiados de manera eficiente.

Las mediciones de rendimiento, aunque con cierta variabilidad en los datos brutos, refuerzan los principios teóricos de la criptografía: las operaciones asimétricas (firma RSA) son significativamente más costosas que las operaciones simétricas (cifrado AES) y hash con llave (HMAC). El HMAC resultó ser, como se esperaba, la operación más rápida, ideal para verificación frecuente de mensajes.

El análisis de los datos de rendimiento bajo diferentes cargas (secuencial vs. concurrente) es crucial para entender el impacto de la seguridad en sistemas escalables. Mientras que las operaciones individuales de un delegado pueden no variar drásticamente con la concurrencia (si el procesador no está saturado), el trabajo *total* del servidor aumenta linealmente con el número de clientes concurrentes, destacando la importancia de la eficiencia algorítmica para manejar altas cargas.

La estimación de la velocidad del procesador, calculada a partir de los tiempos medidos, proporciona una cuantificación de la capacidad de la máquina para realizar estas operaciones criptográficas, aunque los resultados para AES vs RSA en el cifrado de datos específicos medidos son atípicos y sugieren la importancia del tamaño de los datos y la variabilidad de la medición en micro-benchmarks.

10. Referencias

- Stallings, W. (2017). *Cryptography and Network Security: Principles and Practice* (7ª ed.). Pearson Education. ISBN: 9780134444284.
- Tanenbaum, A. S., & Wetherall, D. J. (2010). *Computer Networks* (5ª ed.). Pearson. ISBN: 9780132126953. (Capítulos 7 y 8 tratan de la capa de aplicación y la seguridad).
- Schneier, B. (n.d.). *Blowfish*. Recuperado de https://www.schneier.com/academic/blowfish/
- RSA Security LLC. (n.d.). *RSA Algorithm*. Recuperado de https://www.rsa.com/en-us/company/rsa-labs
- Housley, R., Ford, W., Polk, W., & Solo, D. (2008). *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. RFC 5280. IETF. Disponible en: https://datatracker.ietf.org/doc/html/rfc5280
- Rivest, R. (1992). *The MD5 Message-Digest Algorithm*. RFC 1321. IETF. Disponible en: https://datatracker.ietf.org/doc/html/rfc1321