Diseño e implementación de un sistema colaborativo de verificación de datos segmentados en la nube

C.Angulo-Domínguez
CINVESTAV Unidad Guadalajara
Zapopan 45019, México
cecilia.angulo@cinvestav.mx

A.Díaz-Pérez
CINVESTAV Unidad Guadalajara
Zapopan 45019, México
adiaz@cinvestav.mx

J.L.González-Compeán
CINVESTAV Unidad Tamaulipas
Ciudad Victoria 87130, México
joseluis.gonzalez@cinvestav.mx

***Resumen*—En la actualidad, el almacenamiento de información en la nube se ha convertido en una práctica común tanto en entornos personales como empresariales. No obstante, esta modalidad implica una delegación de la custodia de los datos a terceros, lo que plantea desafíos importantes en términos de confianza, integridad y seguridad. Una de las principales preocupaciones es la posibilidad de que los datos almacenados puedan ser alterados o eliminados sin conocimiento del propietario, lo cual compromete no solo la disponibilidad, sino también la veracidad de la información. Para enfrentar esta problemática, se ha desarrollado el esquema Provable Data Possession (PDP), el cual permite verificar que los datos almacenados en un servidor remoto se mantienen íntegros sin necesidad de descargarlos en su totalidad. En este artículo se propone la implementación de una arquitectura escalable del esquema PDP dentro de una infraestructura basada en servicios de Amazon Web Services (AWS). La propuesta incluye tres componentes principales: el cliente, el servidor y el verificador. Este trabajo se encuentra en una etapa preliminar de consolidación, por lo tanto, los resultados presentados corresponden principalmente a la validación funcional del sistema y en las proyecciones de rendimiento que se esperan alcanzar en pruebas experimentales en etapas posteriores.**

***Palabras clave*—provable data possession, integridad de datos, seguridad en la nube, arquitectura escalable, computación en la nube, verificación remota**

I. Introducción

Con el auge de los servicios de computación en la nube, organizaciones y usuarios individuales han adoptado soluciones de almacenamiento remoto como una alternativa flexible, escalable y económica frente a los métodos tradicionales. Sin embargo, esta delegación del almacenamiento a servicios externos conlleva riesgos inherentes relacionados con la pérdida de control sobre los datos, entre ellos, la manipulación no autorizada, la eliminación maliciosa o accidental, y la falta de garantías sobre la persistencia e integridad de la información [1].

En respuesta a estas inquietudes, se han propuesto diversos mecanismos criptográficos que permiten garantizar la integridad de los datos almacenados [2]–[4]. Uno de los más representativos es el esquema de Provable Data Possession (PDP), que permite al propietario verificar, de manera eficiente, que sus datos permanecen íntegros en un servidor remoto, sin requerir la descarga completa de los archivos. Este esquema se basa en la generación de pruebas criptográficas que el servidor debe responder correctamente, demostrando así que aún conserva los datos originales [5].

La arquitectura planteada busca mostrar la viabilidad de incorporar mecanismos de verificación de integridad de datos en entornos de nube de manera eficiente y escalable. Se enfatiza la importancia de la distribución de funciones en distintos nodos y del aprovechamiento de servicios administrados, lo que simplifica tanto el despliegue como la operación del sistema. La propuesta no solo se presenta como una alternativa a los modelos tradicionales de almacenamiento, sino también como una base para el desarrollo de esquemas auditables y seguros orientados a entornos distribuidos.

Además, la propuesta tiene aplicaciones en escenarios industriales donde la confiabilidad de los datos es crítica. Ejemplos incluyen la trazabilidad en cadenas de suministro, la protección de gemelos digitales en manufactura avanzada, la gestión segura de historiales clínicos en entornos de salud, la verificación de registros en sistemas financieros distribuidos y el resguardo de bitácoras en infraestructuras críticas como energía y transporte. Estos casos de uso evidencian el potencial de la solución para fortalecer la confianza digital en el marco de la Industria.

II. Computación en la nube y provable data possession: Preliminares

En esta sección se presentan los fundamentos conceptuales que sustentan la solución propuesta, abarcando el esquema de Provable Data Possession (PDP), las primitivas criptográficas que lo respaldan y los principios de computación en la nube donde se despliega la arquitectura.

A. Provable Data Possession

El esquema de Provable Data Possession fue introducido formalmente por Ateniese et al. [5] como un mecanismo que permite a un cliente verificar que sus datos se encuentran íntegros en un servidor de almacenamiento remoto, sin necesidad de descargar el archivo completo.

En términos generales, un esquema PDP se basa en un protocolo de desafío y respuesta entre el cliente y el servidor. El cliente divide los archivos en bloques, genera metadatos para cada bloque y los almacena junto con los datos en el servidor. Posteriormente, puede emitir desafíos aleatorios sobre ciertos bloques, y el servidor debe responder con pruebas

válidas de posesión de esos segmentos. Si las respuestas son correctas, se asume con alta probabilidad que los datos permanecen sin alteraciones.

Este enfoque ha sido adaptado y optimizado por diversos trabajos [6]–[8]. Reyes et al. [9] proponen una arquitectura modular basada en building blocks y patrones Manager/Worker, que mejora la eficiencia del PDP mediante contenedores virtuales desplegados en paralelo, permitiendo realizar verificaciones concurrentes en escenarios reales, como el almacenamiento de imágenes satelitales.

B. Funciones Hash Criptográficas

Una función hash criptográfica es un algoritmo que transforma una entrada arbitraria en una salida de longitud fija. Las funciones hash poseen propiedades fundamentales como [10]:

- Resistencia a colisiones: es extremadamente difícil hallar dos entradas diferentes que generen el mismo valor hash. Estas propiedades garantizan la integridad y autenticidad de los datos almacenados.
- Resistencia a preimagen: resulta prácticamente imposible encontrar una entrada que produzca un valor hash dado.
- Resistencia a segunda preimagen: dadas dos entradas distintas, es inviable encontrar otra entrada que genere el mismo hash que una entrada específica.
- Determinismo y eficiencia computacional.

C. Firmas Digitales RSA

Las firmas digitales garantizan la autenticidad y no repudio de los datos transmitidos. En el contexto de PDP, su funcionamiento se puede resumir en:

- Generación de un par de claves: privada (sk) y Pública (pk)
- Firma de los datos (o sus hashes) con la clave privada.
- Verificación de la firma mediante la clave pública.

Cada segmento del archivo es firmado digitalmente antes de ser enviado al servidor. Posteriormente, durante una verificación, el servidor debe proporcionar el segmento firmado, y el verificador podrá comprobar su autenticidad utilizando la clave pública correspondiente. Esta técnica asegura que los datos no han sido modificados y que provienen de una fuente confiable.

Esta técnica también ha sido utilizada en trabajos como el de Li et al. [11], quienes proponen IntegrityChain, una extensión de PDP para entornos descentralizados con blockchain, donde las firmas se combinan con mecanismos de consenso para validar la integridad sin necesidad de confianza en una única entidad.

D. Modelo de Computación en la Nube

El paradigma de la computación en la nube ha transformado significativamente la manera en que las organizaciones acceden, gestionan y escalan sus recursos computacionales. Este modelo permite el acceso remoto, bajo demanda y altamente escalable a servicios de infraestructura, almacenamiento y procesamiento, lo cual reduce costos operativos y mejora la eficiencia del despliegue de sistemas distribuidos. En este contexto, plataformas de nube como Amazon Web Services (AWS) ofrecen un conjunto robusto de servicios que permiten la construcción flexible de arquitecturas orientadas a la seguridad, alta disponibilidad y rendimiento [12]. Entre los servicios para entornos de procesamiento distribuido se destacan:

- EC2: proporciona capacidad de cómputo escalable en la nube, permitiendo el despliegue de máquinas virtuales configurables.
- EFS y S3: permiten el almacenamiento de objetos y archivos de manera persistente y compartida. Mientras que S3 es ideal para almacenar grandes volúmenes de datos no estructurados con alta durabilidad, EFS permite el acceso simultáneo a archivos por múltiples instancias EC2, facilitando la cooperación entre componentes distribuidos.
- IAM: proporciona un control granular sobre el acceso a los recursos de AWS, mediante la definición de políticas y roles. Esto garantiza que solo los agentes autorizados puedan interactuar con los servicios, fortaleciendo así la seguridad del sistema.

El uso coordinado de estos servicios facilita la implementación de sistemas seguros y escalables en entornos cloud, especialmente en arquitecturas donde intervienen múltiples actores y procesos sensibles a la integridad y confidencialidad de la información.

III. Diseño e implementación del sistema PDP

Esta sección describe el diseño e implementación de los componentes que conforman el sistema basado en el esquema Provable Data Possession, desplegado sobre infraestructura en la nube. El sistema está dividido en tres entidades principales: cliente, servidor y verificador, cada una con funciones específicas que en conjunto permiten asegurar la integridad de los datos almacenados.

A. Cliente

El Cliente se desempeña como la entidad propietaria de los datos y es responsable de preparar la información antes de su almacenamiento remoto. Para ello, ejecuta localmente una serie de operaciones previas al envío, tales como la fragmentación de archivos, la generación de metadatos y la aplicación de funciones criptográficas. Estas tareas aseguran que los datos estén correctamente estructurados y protegidos antes de ser transferidos al Servidor. Entre sus funciones principales se encuentran:

- Generar un par de claves criptográficas RSA (clave pública y privada).
- Dividir los archivos en segmentos.
- Calcular el hash SHA-256 de cada segmento.
- Firmar digitalmente cada segmento utilizando RSA.
- Enviar al servidor los segmentos junto con su firma y metadatos asociados.
- Almacenar localmente los metadatos
- Solicitar la verificación de integridad de los segmentos.

En la Figura 1 se representa gráficamente el flujo de operaciones ejecutadas por el cliente. Se puede observar cómo,

tras la segmentación del archivo, cada fragmento se somete a un proceso de hash y firma.

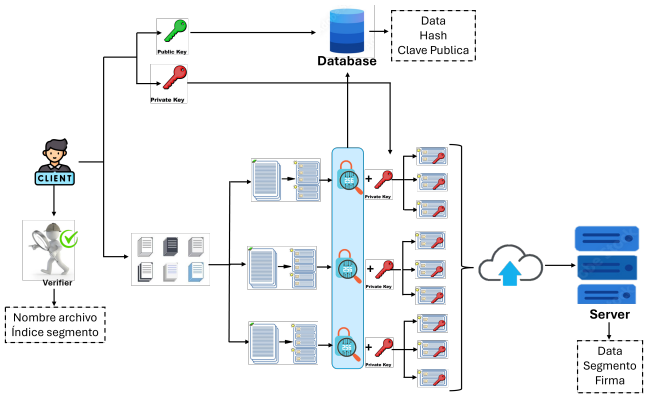

Fig. 1. Representación del componente cliente.

El algoritmo 1 detalla el procedimiento que asegura que cada fragmento enviado al servidor puede ser autenticado posteriormente mediante su firma y verificado con su hash.

Algorithm 1 Proceso del Cliente

$cPrivada, cPublica \leftarrow$ GENERACLAVESRSA()
$BD \leftarrow$ CONECTARSQLITE()
for cada archivo en la lista **do**
  $segmentos \leftarrow$ DIVIDIRARCHIVO($archivo$)
  for cada segmento en segmentos **do**
    $hash \leftarrow$ SHA256($segmento$)
    $firma \leftarrow$ FIRMAR($cPrivada, segmento$)
    ALMACENAMETADATO($data, hash, cPublica$)
    ENVIARALSERVIDOR($data, segmento, firma$)
  end for
end for
SOLICITARVERIFICACION($data$)

B. Servidor

El servidor es responsable de recibir los datos del cliente y almacenarlos de manera segura. Para cada segmento recibido, conserva tanto el contenido como sus metadatos, permitiendo posteriormente su recuperación para procesos de verificación. El servidor también actúa como intermediario entre el cliente y el verificador, respondiendo a solicitudes de verificación mediante la entrega de los segmentos requeridos y sus respectivos metadatos. Sus funciones principales incluyen:

- Recibir los segmentos firmados desde el cliente.
- Asociar cada segmento con su metadato.
- Almacenar los datos de manera organizada.
- Atender solicitudes del verificador enviando los segmentos requeridos junto con sus metadatos.

En la Figura 2 se ilustra el comportamiento del servidor. Se muestra cómo recibe de forma continua los paquetes enviados por el cliente y los almacena, manteniéndolos disponibles para responder las peticiones del verificador.

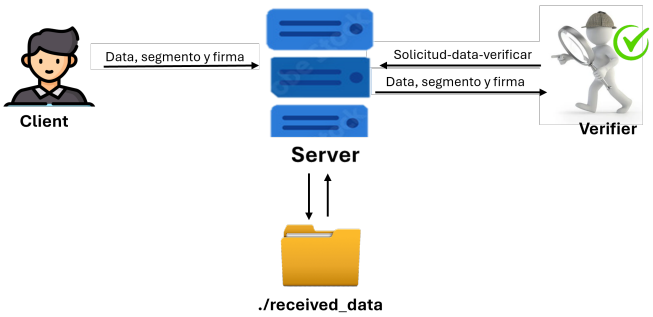

Fig. 2. Representación del componente servidor.

En el Algoritmo 2, el servidor desempeña el rol de un repositorio pasivo de datos, encargado únicamente de almacenar los segmentos recibidos, carece de la capacidad para alterar la información sin ser detectado, ya que cualquier modificación comprometería la verificación de integridad realizada posteriormente por el verificador.

Algorithm 2 Proceso del Servidor

while Servidor activo **do**
  $(origen, peticion) \leftarrow$ ESPERARPETICION()
  if $origen =$ Cliente **then**
    $(data, segmento, firma) \leftarrow$ peticionPost
    ALMACENAR($data, segmento, firma$)
  else if $origen =$ Verificador **then**
    $(data, segmento, firma) \leftarrow$ peticionGet
    ENVIARVERIFICADOR($data, segmento, firma$)
  end if
end while

C. Verificador

El verificador es una entidad encargada de comprobar la autenticidad e integridad de los segmentos almacenados en el servidor. Este proceso se inicia a petición del cliente, quien determina qué bloques deben ser verificados. El verificador solicita al servidor los segmentos correspondientes y, una vez recibidos, valida sus firmas digitales utilizando la clave pública del cliente. Posteriormente, calcula los valores hash de los segmentos y los compara con los valores de referencia previamente almacenados, asegurando que los datos no hayan sido alterados. Las funciones principales del verificador son:

- Solicitar los segmentos designados por el cliente.
- Verificar las firmas digitales empleando la clave pública del cliente.
- Calcular y comparar los hashes para validar la integridad de los datos.

La Figura 3 representa gráficamente el ciclo de verificación, desde la generación del desafío hasta la validación de la integridad de los datos. La figura 3 evidencia cómo el verificador accede a un subconjunto de segmentos sin necesidad de recuperar todo el archivo, reforzando así la eficiencia del esquema PDP.

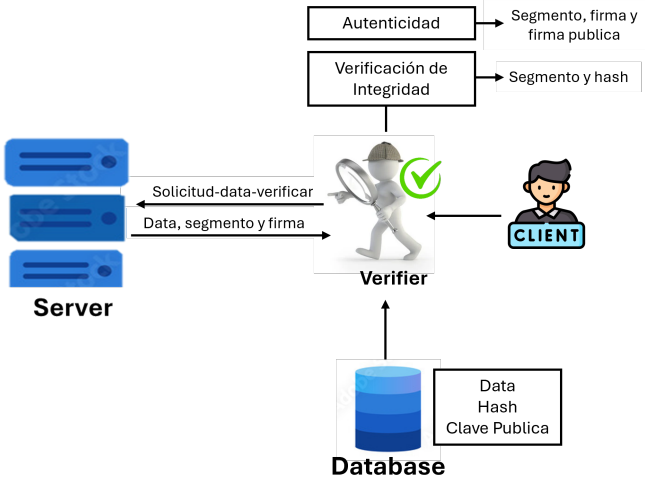

Fig. 3. Representación del componente verificador.

El Algoritmo 3 valida tanto la autenticidad como la integridad de los datos recibidos. Mediante el uso de firma digital y hash garantiza que incluso una mínima alteración en los segmentos será detectada con alta probabilidad.

Algorithm 3 Proceso del Verificador

```
segmentos ← PETICIONCLIENTE()
segmentosRecib ← SOLICITARSEGMENTOSSERVIDOR()
for cada segmento recibido do
    autenticidad ← VERIFICAFIRMA(segmento, firma)
    integridad ← VERIFICAHASH(segmento, hash)
end for
```

IV. Arquitectura del sistema en la nube

La arquitectura propuesta para la implementación del esquema Provable Data Possession se despliega sobre una infraestructura en la nube utilizando servicios de Amazon Web Services. Esta arquitectura se diseñó con base en principios de modularidad y escalabilidad, permitiendo separar los componentes funcionales del sistema (cliente, servidor y verificador) en diferentes instancias o servicios, y facilitar así su despliegue, mantenimiento y evolución.

A. Descripción general de la arquitectura

La arquitectura se compone de tres entidades lógicas principales:

- Cliente: desplegado en una instancia EC2, se encarga de realizar la segmentación de los datos, generar las firmas digitales correspondientes y transferir los segmentos al servidor.
- Servidor: implementado sobre una instancia EC2, actúa como repositorio de los datos segmentados. Almacena los bloques firmados y responde a las solicitudes del verificador sin capacidad de modificar los datos sin ser detectado.
- Verificador: ejecutado en una instancia EC2, recibe del cliente la instrucción de verificar determinados segmentos. Solicita dichos segmentos al servidor y valida su autenticidad mediante la verificación de firmas y el cálculo de hashes.

B. Diagrama de la arquitectura en la nube

El sistema se estructura en tres niveles funcionales (ver Figura 4), cada uno correspondiente a uno de los roles del esquema Provable Data Possession (PDP). La infraestructura ha sido desplegada sobre servicios de Amazon Web Services (AWS), utilizando Terraform como herramienta de aprovisionamiento y automatización. Esta elección permite gestionar la infraestructura como código, garantizando la reproducibilidad, escalabilidad y trazabilidad de las configuraciones, además de facilitar el control de versiones y la gestión de cambios en el entorno. El flujo de operación se describen a continuación:

- Inicialización:
  - El cliente genera su propio par de claves RSA para firmar sus datos.
  - El servidor permanece disponible y expone un endpoint para recibir segmentos y metadatos.
- Carga de datos:
  - El cliente divide el archivo en segmentos, calcula sus hashes, firma cada uno y envía los segmentos y algunos metadatos firmados al servidor, mientras que los hashes y metadatos se guardan en un sistema de almacenamiento compartido (EFS) accesible por clientes y verificadores.
  - El servidor almacena los segmentos de datos y metadatos recibidos por el cliente localmente.
- Verificación:
  - A solicitud del cliente, el verificador inicia un proceso de validación sobre un subconjunto aleatorio de segmentos.
  - El verificador solicita dichos segmentos al servidor, quien los recupera desde su almacenamiento local.
  - Utilizando las firmas y hashes almacenados en EFS, el verificador comprueba la integridad y autenticidad de los datos.
  - Los resultados de la verificación se registran, y en caso de detectar inconsistencias, se generan alertas automáticas.
- Escalabilidad y replicación:
  - Varios clientes y verificadores pueden operar simultáneamente sobre la infraestructura compartida, permitiendo múltiples procesos de carga, consulta y verificación en paralelo sin interferencias entre sí.
  - Las instancias EC2 del servidor pueden escalar horizontalmente según la demanda.
  - La replicación de datos en S3 y EFS garantiza la disponibilidad ante fallos de zona o región, asegurando la continuidad del servicio y la integridad de la información almacenada.

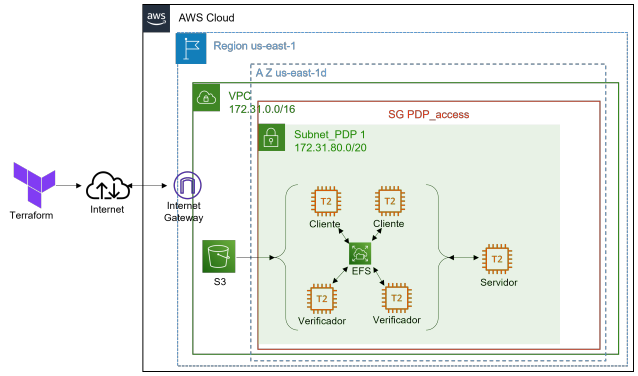

Fig. 4. Arquitectura del esquema PDP en la nube.

V. Resultados esperados y pruebas de concepto

La presente sección tiene como objetivo exponer los resultados esperados derivados de la implementación del esquema Provable Data Possession en un entorno de computación en la nube, así como describir las pruebas de concepto iniciales realizadas para validar su funcionamiento a nivel lógico, criptográfico y operativo. Dado que este trabajo se encuentra en fase de consolidación arquitectónica, los resultados aquí descritos se enfocan en la validación funcional del sistema y en las proyecciones de rendimiento que se esperan alcanzar en pruebas experimentales futuras.

A. Resultados esperados

El sistema propuesto fue diseñado bajo principios de seguridad y modularidad. En este sentido, los resultados esperados se agrupan en las siguientes categorías:

- Seguridad de los datos.
  - Integridad garantizada: gracias al uso de funciones hash SHA-256 y firmas digitales RSA, se espera que cualquier alteración en los datos pueda ser detectada con alta certeza, incluso al verificar solo un subconjunto aleatorio de segmentos.
  - Autenticidad de origen: la verificación mediante la clave pública permite confirmar que los datos provienen efectivamente del cliente autorizado.
  - Sin necesidad de copia local: el cliente puede eliminar los archivos originales una vez almacenados, manteniendo solo los metadatos.
- Eficiencia y escalabilidad
  - Bajo consumo de ancho de banda: el sistema permite verificar archivos completos mediante el análisis de unos pocos bloques, evitando transferencias masivas de datos.
  - Escalabilidad horizontal: el sistema está diseñado para permitir múltiples clientes y verificadores operando en paralelo sin interferencias.
  - Procesamiento distribuido: al separar los roles funcionales en instancias EC2 distintas, se espera un aprovechamiento más equilibrado de los recursos computacionales.

B. Pruebas de concepto realizadas

Se llevaron a cabo pruebas de concepto sobre la arquitectura implementada en AWS con el fin de verificar la coherencia entre los módulos y la viabilidad del flujo operativo. Estas pruebas no se enfocaron en rendimiento cuantitativo, sino en la validación funcional.

Prueba 1: Flujo completo de un archivo simple.
Objetivo: verificar que el ciclo completo(cliente, servidor, verificador) se realiza correctamente con un archivo de prueba.

- Acciones realizadas:
  - Segmentación y firma de un archivo de 5 MB en el cliente.
  - Almacenamiento del archivo segmentado en el servidor.
  - Solicitud del cliente al verificador la comprobación de 3 bloques.
  - Solicitud de obtención de 3 bloques del verificador al servidor para su validación.
  - Verificación exitosa de firmas y hashes.

Resultado: flujo completado sin errores y con tiempos de respuesta aceptables ($< 2\,$s por segmento).

Prueba 2: Detección de corrupción intencionada.
Objetivo: comprobar la eficacia del sistema ante la modificación maliciosa de un bloque.

- Acciones realizadas:
  - Se alteró manualmente uno de los segmentos en el almacenamiento del servidor.
  - El verificador solicitó el bloque modificado.
  - La verificación de firma y hash falló.

Resultado: el sistema detectó correctamente la modificación, confirmando la funcionalidad de las garantías de integridad.

Prueba 3: Operación concurrente básica.
Objetivo: simular la operación de múltiples clientes/verificadores en paralelo.

- Acciones realizadas:
  - Dos clientes ejecutaron simultáneamente envíos de archivos distintos.
  - Un verificador consultó bloques de ambos conjuntos en paralelo.
  - Las respuestas fueron gestionadas correctamente por el servidor.

Resultado: la arquitectura soportó la concurrencia básica sin colisiones ni errores, validando su diseño modular.

Aunque las pruebas de concepto validaron la funcionalidad del sistema, su alcance sigue siendo limitado. Será necesario en futuras fases incorporar métricas de desempeño más detalladas, tiempos de verificación, latencia, uso de recursos, ancho de banda y costos en AWS, así como ensayos de escalabilidad con múltiples clientes, archivos de gran tamaño y condiciones de red diversas.

VI. Conclusiones y trabajos futuros

En esta sección se presentan las conclusiones derivadas del desarrollo e implementación del sistema propuesto, así como una reflexión sobre sus aportaciones técnicas.

En este trabajo se presentó el diseño e implementación de un sistema basado en el esquema Provable Data Possession, desplegado sobre una arquitectura modular en la nube utilizando servicios de AWS. El objetivo principal del sistema es permitir la verificación eficiente de la integridad de los datos almacenados en servidores remotos, sin requerir su descarga completa, utilizando mecanismos criptográficos como funciones hash y firmas digitales.

La propuesta se estructuró en tres componentes principales: cliente, servidor y verificador, los cuales interactúan de forma autónoma y distribuida para garantizar la seguridad y disponibilidad de los datos. Las pruebas de concepto realizadas permitieron validar el flujo operativo, la detección de modificaciones en los datos y la correcta separación de responsabilidades entre componentes.

Entre los beneficios más relevantes de la solución se encuentran su escalabilidad horizontal, la eficiencia en el uso de recursos, y la posibilidad de adaptarse a distintos entornos y políticas de seguridad. Además, la arquitectura propuesta proporciona una base sólida para extender el sistema hacia entornos más complejos o regulados.

Un aspecto crítico identificado en esta fase es la gestión de claves criptográficas. Actualmente, la generación y resguardo de las claves RSA recae en el cliente, lo cual implica riesgos en caso de pérdida, compromiso o mal manejo. La ausencia de mecanismos de rotación de claves, recuperación segura y almacenamiento en módulos de seguridad hardware (HSM) representa una vulnerabilidad que deberá ser atendida en el futuro para asegurar la solidez del sistema.

La propuesta contribuye directamente a los pilares de la Industria 5.0 al fortalecer la confianza digital en servicios de almacenamiento remoto, incrementar la resiliencia frente a ataques mediante verificaciones criptográficas distribuidas, y favorecer la soberanía tecnológica al garantizar que los datos permanezcan bajo control verificable del cliente, incluso cuando son gestionados por proveedores externos.

Trabajos futuros

Como parte del trabajo a futuro, se contemplan diversas líneas de mejora e investigación:

- Evaluación experimental a gran escala: se proyecta realizar pruebas sistemáticas en escenarios con múltiples clientes y verificadores concurrentes, manejo de archivos de gran tamaño (del orden de gigabytes o terabytes), y despliegues distribuidos en nodos heterogéneos dentro de Kubernetes y diferentes regiones de nube.
- Mejora de la gestión de claves: se planea incorporar mecanismos de rotación de claves, recuperación segura y uso de módulos de seguridad hardware (HSM), con el fin de mitigar riesgos derivados del mal manejo o compromiso de las claves criptográficas.
- Se explorará la aplicación de Zero-Knowledge Proofs(ZKP) para mejorar la privacidad en la validación de datos.
- Incorporación de algoritmos alternativos: se analizará el uso de esquemas criptográficos más ligeros, como ECC y BLS, que podrían reducir la sobrecarga computacional frente al uso de RSA y SHA-256 en escenarios de alta concurrencia.
- Evaluación en entornos regulados: se plantea validar el sistema bajo normativas internacionales de ciberseguridad en la nube, tales como ISO/IEC 27017, ISO/IEC 27018, la directiva NIS2 y las guías de seguridad del NIST.

Referencias

[1] K. Zhu, Y. Ren, and Q. Zhu, "A provable data possession protocol in cloud storage systems with fault tolerance," in *2021 IEEE Conference on Dependable and Secure Computing (DSC)*, 2021, pp. 1–6.

[2] H. Yan, J. Li, and Y. Zhang, "Remote data checking with a designated verifier in cloud storage," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1788–1797, 2020.

[3] N. Kaaniche, M. Laurent, and S. Canard, "Cooperative set homomorphic proofs for data possession checking in clouds," *IEEE Transactions on Cloud Computing*, vol. 9, no. 1, pp. 102–117, 2021.

[4] M. Tian, Z. Xie, H. Zhong, and Z. Chen, "Lightweight proofs of storage with public verifiability from lattices," in *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2020, pp. 551–558.

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 598–609. [Online]. Available: https://doi.org/10.1145/1315245.1315318

[6] S. OGISO, M. MOHRI, and Y. SHIRAISHI, "Transparent provable data possession scheme for cloud storage," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, 2020, pp. 1–5.

[7] S. Dong, J. Jiao, and S. Li, "A multiple-replica provable data possession algorithm based on branch authentication tree," in *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, 2020, pp. 400–404.

[8] X. A. Wang and R. Li, "Improved secure provable data possession scheme for cloud storage," in *2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, 2022, pp. 456–460.

[9] H. G. Reyes-Anastacio, J. Gonzalez-Compean, M. Morales-Sandoval, and J. Carretero, "A data integrity verification service for cloud storage based on building blocks," in *2018 8th International Conference on Computer Science and Information Technology (CSIT)*, 2018, pp. 201–206.

[10] V. Mulder, A. Mermoud, V. Lenders, and B. Tellenbach, *Hash Functions*. Springer Nature, 2023.

[11] Y. Li, Y. Yu, R. Chen, X. Du, and M. Guizani, "Integritychain: Provable data possession for decentralized storage," *IEEE Journal on Selected Areas in Communications*, vol. PP, pp. 1–1, 04 2020.

[12] S. Shrivastava, N. Srivastav, A. Artasanchez, I. Sayed, and S. D, *AWS for Solutions Architects: The definitive guide to AWS Solutions Architecture for migrating to, building, scaling, and succeeding in the cloud*. Packt Publishing, 2023. [Online]. Available: https://books.google.com.mx/books?id=x6W7EAAAQBAJ