Julián Oliveros Forero – 201821595

Álvaro Plata – 201820098

CASO 3 - INFRAESTRUCTRA COMPUTACIONAL

Tabla de contenido

Índice de tablas y figuras

1. Ejecución del programa en diferentes escenarios

Para ejecutar nuestra solución y los escenarios de prueba definidos, utilizamos las siguientes infraestructuras:

Tabla 1: Especificaciones de las infraestructuras

Computador	MacBook Air 2015	Windows
Threads	16	32
CPU	Intel Core i5-5250U 1.60GHz	AMD Ryzen 9 4900HS with Radeon Graphics
SO	macOS Big Sur	Windows 10 Pro
Cores	2	8
RAM	8 GB	16

Según las pruebas realizadas, observamos que el tiempo promedio para generar todas las posibles sales es de 10 a 14 horas. Debido al tiempo utilizado en las pruebas de 32 ceros, y que algunas de ellas arrojaron como resultado que no se encontró una sal que generara un hash que cumpliera la condición, por lo que tuvieron que evaluar todas las posibilidades, no fue posible la terminación de las pruebas con 36 ceros. Sin embargo, al estar ejecutándolas, pudimos notar que el tiempo que estaban tomando era muy similar al de las pruebas con 32 ceros. Además, consideramos que la probabilidad de existe una cadena que inicie con 36 ceros (9 ceros si esta en hexadecimal) es muy baja: del 2.8x10^-12 %, por lo que también se puede dar el caso de que se pruebe con todas las sales posibles y no se encuentra un hash que cumpla con los criterios mencionados anteriormente.

1.1 **Algoritmo SHA-256**

Tabla 2: Resultados de las pruebas en Windows con el algoritmo SHA-256, de 20 a 24 ceros

ALGORITMO SHA-256				
Bits en 0\ Cadena		**buildtdarchs**	**Infraestructura computacional**	**uniandes**
20	Sal	aqcze	ajqcp	apnxg
	Hash	000008318cc425329223beb17d03f32694db79d47f94596c5e04880c027c4a86	000006bb99d3d9102601a5c91fbabcabf2ec1ddf0eaafcf55e62f52134f8de1c	0000056b182fa729dc75b5888b32c9c0c7e4196c102bb0ead9000cbdc794d9f7
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	0,0194	0,0128	0,01455
24	Sal	bbxpcw	ccujmr	amepki
	Hash	000000e463b6fc2a5822bfae4e40ff70a5a593947d0c20b586e4a73db18beaaa	000000c00bbbc19e559a619fe8c18b8814afd0cab6fea9158451901e5fa13853	0000005233b4961347a26ca647def0d35cb6da259372d1770df91af2aa8a1db7
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	0,509533333	0,7574	0,370316667

Tabla 3: Resultados de las pruebas en Windows con el algoritmo SHA-512, de 28 a 32 ceros

28	Sal	amgxevn	hjcoza	pwvezf
	Hash	00000005dfbc4dfb5906839ba6e20eda823ca02849e9c73d3dc1e4d0d08bfd92	0000000da869c4d01369952cf7a1db51faf3027bacc0c1558c67027df7609057	000000099440bec4d964c3ae117c90ff792697d52573c3542401de1bd72b663b
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	9,57015	2,068283333	4,156083333
32	Sal	fvsuohu	jzwgxuf	plonjna
	Hash	000000005ce0913c89986b2d668c452c776137b2f16d56ffd0bc8d07284b3c9b	00000000768803555617f37bc74b8140f7b5cf0c1e63acdf4f7f4e60a26e593	000000001bc28a870d206cc4c62c0fbd538fc5e633fc314eaf626f8991db5e63
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	44,04226667	68,41981667	102,89605

Tabla 4: Resultados de las pruebas en Mac con el algoritmo SHA-256, de 20 a 24 ceros

ALGORITMO SHA-256				
Bits en 0\ Cadena		**buildtdarchs**	**Infraestructura computacional**	**uniandes**
20	Sal	aqcze	ajqcp	apnxg
	Hash	000008318cc425329223beb17d03f32694db79d47f94596c5e04880c027c4a86	000006bb99d3d9102601a5c91fbabcabf2ec1ddf0eaafcf55e62f52134f8de1c	0000056b182fa729dc75b5888b32c9c0c7e4196c102bb0ead9000cbdc794d9f7
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	0,111366667	0,123133333	1,099716667
24	Sal	bbxpcw	ccujmr	amepki
	Hash	000000e463b6fc2a5822bfae4e40ff70a5a593947d0c20b586e4a73db18beaaa	000000c00bbbc19e559a619fe8c18b8814afd0cab6fea9158451901e5fa13853	0000005233b4961347a26ca647def0d35cb6da259372d1770df91af2aa8a1db7
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	1,6635	2,962533333	1,4375

Tabla 5: Resultados de las pruebas en Mac con el algoritmo SHA-256, de 28 a 32 ceros

28	Sal	amgxevn	hjcoza	pwvezf
	Hash	00000005dfbc4dfb5906839ba6e20eda823ca02849e9c73d3dc1e4d0d08bfd92	0000000da869c4d01369952cf7a1db51faf3027bacc0c1558c67027df7609057	000000099440bec4d964c3ae117c90ff792697d52573c3542401de1bd72b663b
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	32,18028333	6,855783333	14,1411
32	Sal	fvsuohu	jzwgxuf	plonjna
	Hash	000000005ce0913c89986b2d668c452c776137b2f16d56ffd0bc8d07284b3c9b	00000000768803555561 7f37bc74b8140f7b5cf0c1e63acdf4f7f4e60a26e593	000000001bc28a870d206cc4c62c0fbd538fc5e633fc314eaf626f8991db5e63
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	139,5700333	216,821	326,07

1.2 Algoritmo SHA-512

Tabla 6: Resultados de las pruebas en Windows con el algoritmo SHA-512, de 20 a 24 ceros

ALGORITMO SHA-512				
Bits en 0\ Cadena		**buildtdarchs**	**Infraestructura computacional**	**uniandes**
20	Sal	anxbm	etkhs	qsdp
	Hash	000009a79c33030e370dc61d1e393159d69d1d6173f25e17a5578d61c77b31d1c9bcb8c28dde7e9df557c156ee93c0fa3cfdaabf9cfe1a90eeb29e770f55936e	000009dcb11b96160158ca08aca9ccdb2ab7a5d68f80026ed1be5bb0324f94ea5e0ac5af727d474f086fce3d267d17cf23500ac329779336efd0ce7569fe4b85	00000123d51fd86fd0a86c9567d7446f491dab6d9138f5e0478d013c452505f1c5ca0004dedb6d4019ba9a6f6c98ee4d20e30b96eea3d7af4f29d135d431031a
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	0,020633333	0,058283333	0,006616667
24	Sal	ahoamf	ybvzy	dohue
	Hash	0000006222b2402d459b484ce8302b54526637aee0d61e2e762a8ea1912a59b0ba43e7aa7b6a156eb4159847e690ee4b00bcfd261ee7affc5b473333e075cc5d	0000005959ff40b9983499b3ff94f09794d604eed1233e60d2fd74b60518ce92a725d8cfa179ac40ef74bfcabf5fe80ec0f6b30c4382d64c719152762b94b168	000000c1e55728030e3df11f034e78602da6fcf3226afe601b5f7a90eba0b4822447a4f07bc018285e4398b5a68fe65ac75ba34a222afb75f82496ffc929602f
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	0,353333333	0,254866667	0,045033333

Tabla 7: Resultados de las pruebas en Windows con el algoritmo SHA-512, de 28 a 32 ceros

28	Sal	bxhnrdn	bategrc	dywcfb
	Hash	00000005d9f2b05458c655da932d553cd110ad3fcfea2f437aad4b25687bc3cadea518d2e173be15cd50b49743222167544e9b7ac0574f1a59efff1832058b3a	00000004177b4c72e677d5e81e381051b17f1e8df70cf76cfa6f9b5ba01223f579f1ebe517e74f11da7ce21907db468210d46443b3a78171850df2f4d068d17f	00000006573fc76f6b513663eaeb3b5a8d8b561889075372a30c8cac7f8c08cf082ef2ed727ddf6ce81506967990bb384ddd9e89fc4487f7cfd18214d180871f
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	20,57916667	14,5941	1,359766667
32	Sal	cpgappj	cnbsdqa	
	Hash	00000000c2f1b5d47de6d1da83464ef684ed7a65123b7cf257058619e2525c142329c90dab18e311d7662d9222dd3cc071c80939c6e8ec7f43d9aba6dbce9076	00000000d04b59781412bf97cef626e023a1f9046164b252a62466298b0afe6af7fab2ce5e42a8e42aaf023ff5a3a3e4e96b2995e634e12ada621d8e65a25410	NO SE ENCONTRÓ UN HASH QUE CUMPLIERA LA CONDICIÓN
	Computador	Windows	Windows	Windows
	Tiempo (minutos)	25,71831667	25,10951667	720

Tabla 8: Resultados de las pruebas en Mac con el algoritmo SHA-512, de 20 a 24 ceros

ALGORITMO SHA-512				
Bits en 0\ Cadena		**buildtdarchs**	**Infraestructura computacional**	**uniandes**
20	Sal	anxbm	etkhs	qsdp
	Hash	000009a79c33030e370dc61d1e393159d69d1d6173f25e17a5578d61c77b31d1c9bcb8c28dde7e9df557c156ee93c0fa3cfdaabf9cfe1a90eeb29e770f55936e	000009dcb11b96160158ca08aca9ccdb2ab7a5d68f80026ed1be5bb0324f94ea5e0ac5af727d474f086fce3d267d17cf23500ac329779336efd0ce7569fe4b85	00000123d51fd86fd0a86c9567d7446f491dab6d9138f5e0478d013c452505f1c5ca0004dedb6d4019ba9a6f6c98ee4d20e30b96eea3d7af4f29d135d431031a
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	0,821066667	0,359733333	0,032416667
24	Sal	ahoamf	ybvzy	dohue
	Hash	0000006222b2402d459b484ce8302b54526637aee0d61e2e762a8ea1912a59b0ba43e7aa7b6a156eb4159847e690ee4b00bcfd261ee7affc5b473333e075cc5d	0000005959ff40b9983499b3ff94f09794d604eed1233e60d2fd74b60518ce92a725d8cfa179ac40ef74bfcabf5fe80ec0f6b30c4382d64c719152762b94b168	000000c1e55728030e3df11f034e78602da6fcf3226afe601b5f7a90eba0b4822447a4f07bc018285e4398b5a68fe65ac75ba34a222afb75f82496ffc929602f
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	1,6437	1,637733333	0,18745

Tabla 9: Resultados de las pruebas en Mac con el algoritmo SHA-512, de 28 a 32 ceros

28	Sal	bxhnrdn	bategrc	dywcfb
	Hash	00000005d9f2b05458c655da932d553cd110ad3fcfea2f437aad4b25687bc3cadea518d2e173be15cd50b49743222167544e9b7ac0574f1a59efff1832058b3a	00000004177b4c72e677d5e81e381051b17f1e8df70cf76cfa6f9b5ba01223f579f1ebe517e74f11da7ce21907db468210d46443b3a78171850df2f4d068d17f	00000006573fc76f6b513663eaeb3b5a8d8b561889075372a30c8cac7f8c08cf082ef2ed727ddf6ce81506967990bb384ddd9e89fc4487f7cfd18214d180871f
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	103,6666	64,66431667	6,646666667
32	Sal	cpgappj	cnbsdqa	
	Hash	00000000c2f1b5d47de6d1da83464ef684ed7a65123b7cf257058619e2525c142329c90dab18e311d7662d9222dd3cc071c80939c6e8ec7f43d9aba6dbce9076	00000000d04b59781412bf97cef626e023a1f9046164b252a62466298b0afe6af7fab2ce5e42a8e42aaf023ff5a3a3e4e96b2995e634e12ada621d8e65a25410	NO SE ENCONTRÓ UN HASH QUE CUMPLIERA LA CONDICIÓN
	Computador	MacBook Air 2015	MacBook Air 2015	MacBook Air 2015
	Tiempo (minutos)	118,3575167	117,6871667	720

1.3 **Gráficas de resultados de las pruebas:**

En los gráficos podemos evidenciar que, para las pruebas con las 3 cadenas, la gráfica generada por el computador Windows es muy similar a la gráfica generada por el computador MacBook, es decir, que las líneas de los dos algoritmos guardan casi la misma proporción en ambos computadores. Esto se debe a que se está ejecutando el mismo algoritmo en las dos computadoras, sin embargo, podemos ver que las pendientes a medida que se aumenta la cantidad de ceros para el computador Windows son menores que en las del computador MacBook, esto se debe a que la capacidad computacional (se explicará la diferencia en el punto 2) del computador Windows es mayor que la del MacBook.

Tabla 10: Gráfica de resultados para la cadena buildtddarchs en Windows

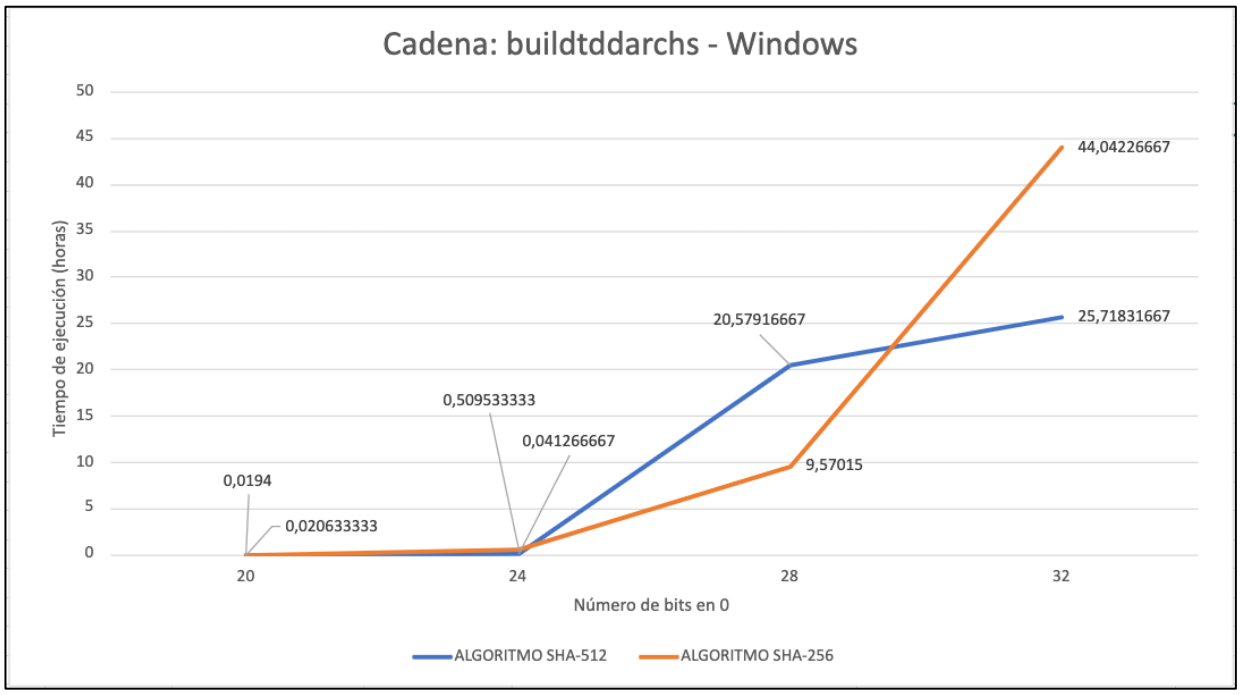


Tabla 11: Gráfica de resultados para la cadena buildtddarchs en Mac

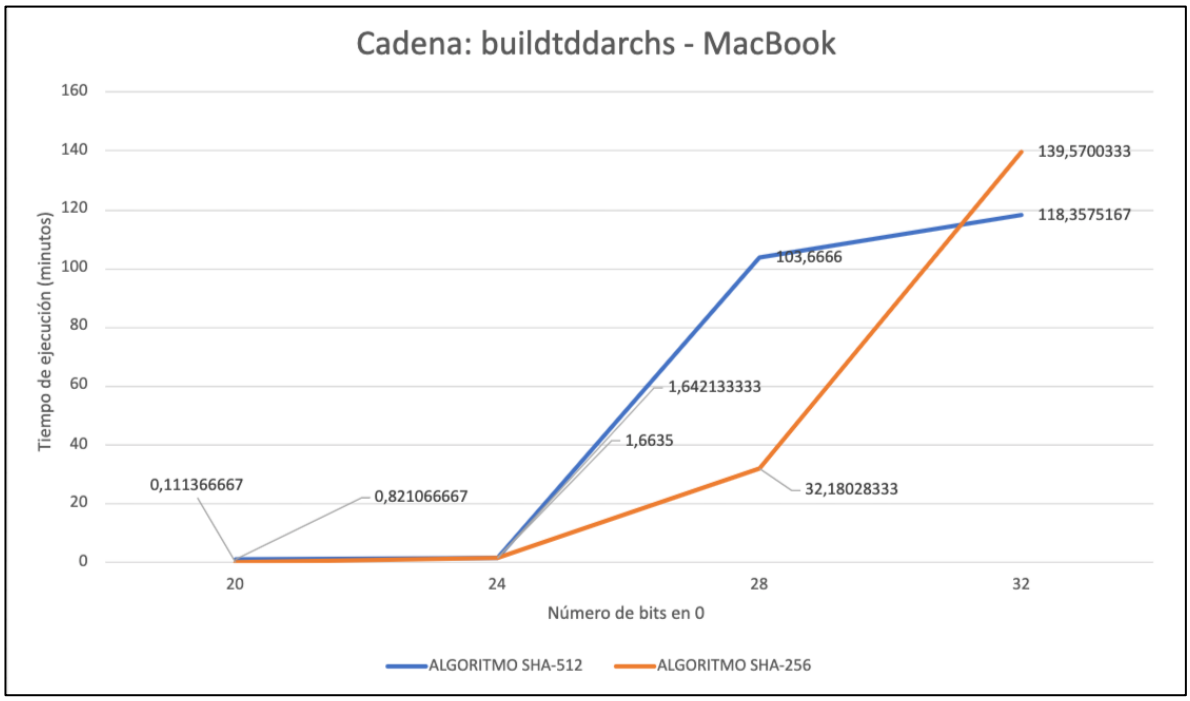


Tabla 12: Gráfica de resultados para la cadena Infraestructura computacional en Windows

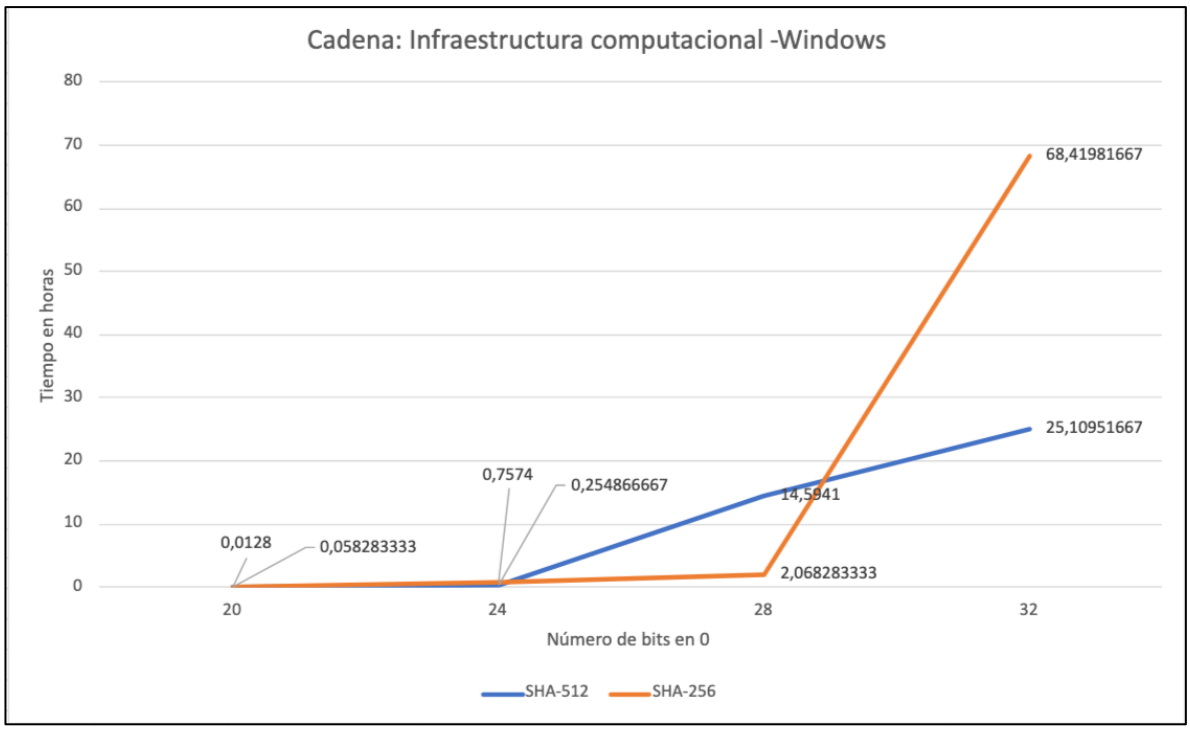


Tabla 13: Gráfica de resultados para la cadena Infraestructura computacional en Mac

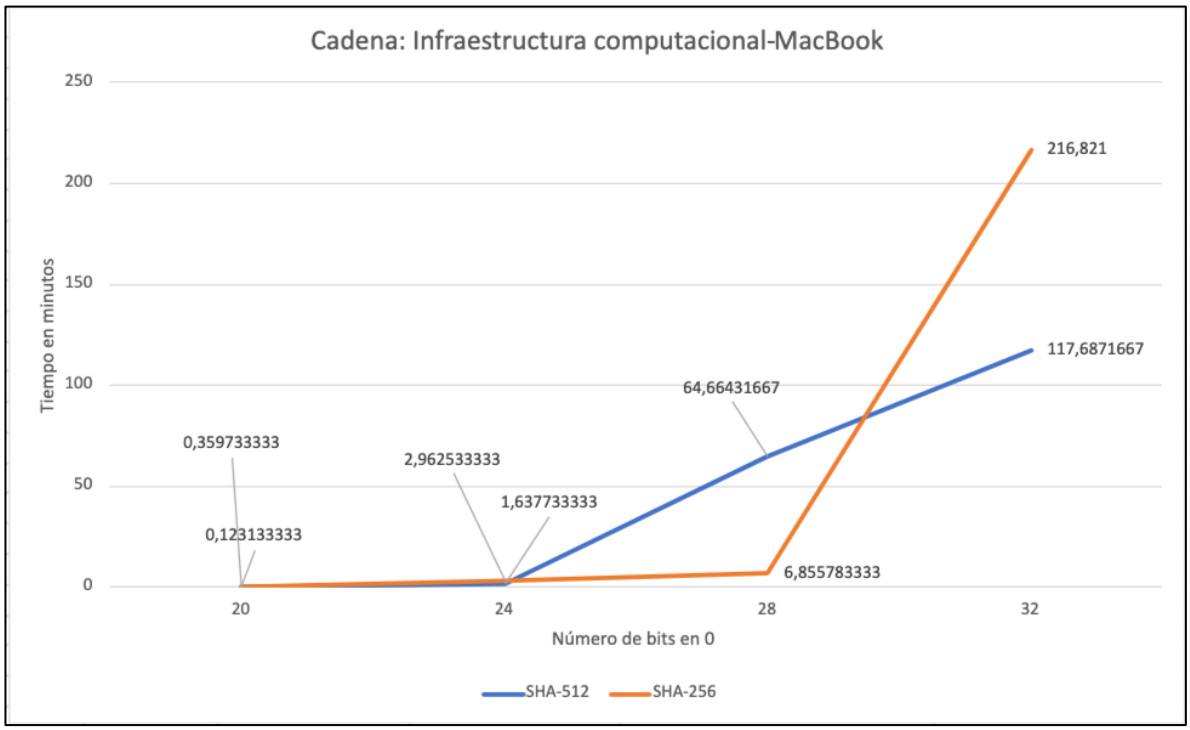


Tabla 14: Gráfica de resultados para la cadena uniandes en Windows

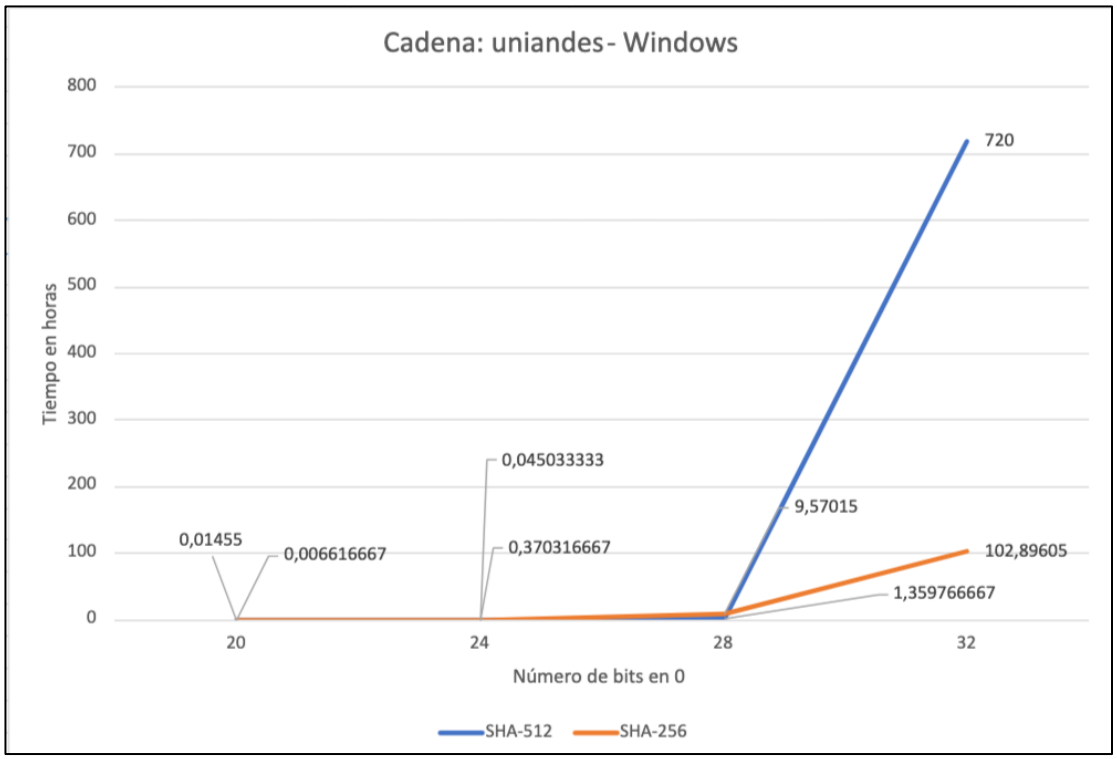
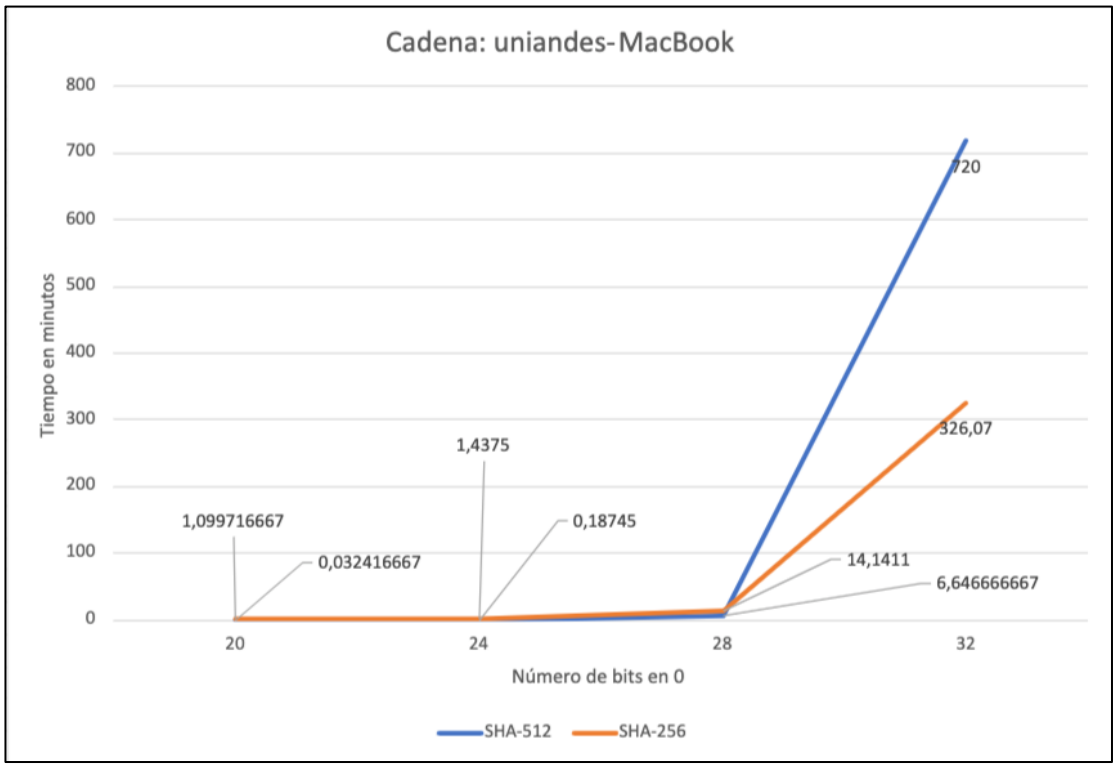


Tabla 15: Gráfica de resultados para la cadena uniandes en Mac



2. Cálculos del procesador

Para calcular el tiempo promedio que toma evaluar un valor y determinar si cumple la condición, hicimos 5 pruebas. Para éstas, modificamos nuestro código para que, al primer hash evaluado, devolviera el tiempo transcurrido y se detuviera.

Los resultados para ambos computadores donde ejecutamos las pruebas fueron los siguientes:

Para MacBook Air:

Velocidad Procesador: 1,6 GHz

Los tiempos en milisegundos para cada prueba fueron: 137, 102, 362,107, 101. Lo que quiere decir que, en promedio, evaluar un valor toma 161.8 milisegundos.

Esto quiere decir que cada ciclo de reloj toma 0.625 nanosegundos, lo que equivale a 0.000000625 milisegundos.

Entonces, podemos concluir que, en promedio, evaluar un valor toma (161.8/0.000000625) = 258,880,000 ciclos de procesador.

Para Windows:

Velocidad Procesador: 3 GHz

Los tiempos en milisegundos para cada prueba fueron: 22, 14, 14 ,14, 13 Lo que quiere decir que, en promedio, evaluar un valor toma 15.4 milisegundos.

Esto quiere decir que cada ciclo de reloj toma 0.33 nanosegundos, lo que equivale a 0.00000033 milisegundos.

Entonces, podemos concluir que, en promedio, evaluar un valor toma (15.4/0.00000033) = 466666667 ciclos de procesador

3. Información de códigos criptográficos

Pudimos encontrar la existencia de aproximadamente 21 códigos de hash, sin embargo, algunos son más usados que otros además de algunos que ya se consideran obsoletos y otros que se siguen considerando como seguros. De estos 21 códigos los más famosos y utilizados son el DES, AES, MD5, SHA (Secure Hash Algorithm), RIPEMD. La entidad que valida la legitimidad de los algoritmos se llama la NIST (National Institute of Standards and Technology).

3.1 **Usados hoy en día.**

Dentro de los algoritmos que se recomiendan usar hoy en día pues se considera que para lograr descifrar por fuerza bruta la repuesta requerirá gran cantidad de poder computacional y tiempo mayor a 1 año se encuentra los siguientes:

- SHA-2: Hay que tener en cuenta que los algoritmos de hash tienen varias variantes como SHA-224, SHA-256, SHA-384, SHA-512. Estas se diferencian de la cantidad de bits que contiene el hash, es decir la salida o respuesta del algoritmo.

El SHA-256 es usado para le seguridad en protocolos de comunicación como SSL, SSH, También es utilizado por Linux y Unix para asegurar las contraseñas con este hash. Sumado a esto el SHA-256 también es usado para muchas blockchains como bitcoin.

- SHA-3: EL SHA-3 fue publicado en el 2015 por la NIST, este puede ser utilizado para sustituir el SHA-2 en caso de ser necesario o si se requiere mejorar la seguridad significativamente.

- AES: Advanced Encryption Standart este es uno de los algoritmos más seguros según afirma la NSA (Nacional Security Agency) "Se basa en varias sustituciones, permutaciones y transformaciones lineales, ejecutadas en bloque de datos de 16 bytes, que se repiten varias veces." Estos algoritmos son usados por bancos, gobiernos y los sistemas de alta seguridad del mundo.

- 3DES: Este algoritmo se basa en la lógica del algoritmo DES, pues este como su nombre lo indica hace un triple cifrado de DES con tres claves distintas porque lo que es más seguro, aunque sigue siendo usado para pagos electrónicos, este se está comenzado a sustituir por el algoritmo AES.

3.2 **Por qué se dejaron de usar los obsoletos**

Principalmente los algoritmos de hash se convierten en obsoletos o inseguros cuando se logran romper, es decir que es posible descifrar cual fue el mensaje que se cifro en un tiempo relativamente corto esto se debe a que cada vez se tiene más poder computacional o por que se encuentran fallos en la lógica matemática de los algoritmos lo que facilita romperlos. Dentro de estos algoritmos que se consideran ya obsoletos están:

- DES: Este algoritmo fue remplazo por el algoritmo AES por lo que el DES quedo obsoleto por que el tamaño de la clave es de 56 bits, el cual como se mencionó anteriormente es muy pequeño para la capacidad de cómputo para el 2000.

- MD4: Este algoritmo se puede considerar obsoleto pues su longitud de 128 bits resulta ser baja en la actualidad, además se encontró que dos entradas podían dar el mismo resultado.

- MD5: Este algoritmo se consideró obsoleto pues dentro de sus vulnerabilidades esta que pueden ocurrir colisiones, es decir se puede producir una misma salida para dos entradas. Según afirma Eleven Paths "algoritmo de cifrado MD5 como vulnerable o "oficialmente roto" data de 2004, cuando Xiaoyun Wang y su equipo anunciaron el descubrimiento de colisiones de hash para MD5."

- SHA1: Este algoritmo fue declarado como roto a inicios del 2017, el motivo es por colisión de hash, el mismo motivo que se identificó para el MD4 y MD5. Según afirma (García, 2017) "los investigadores, este método de colisión es 100.000 veces más rápido que un ataque por fuerza bruta. En total, se utilizaron 9.223.372.036.854.775.808 (9,2 trillones) de computaciones SHA-1. El procedimiento tardaría 110 años en ser realizado con una sola tarjeta gráfica, o puesto de otra forma, 1 año con 110 tarjetas gráficas".

4. Tecnología Blockchain

Un caso de uso que identificamos para blockchain en para el sistema de votación para los países, pues utilizar blockchain para el conteo de votos haría que el proceso fuese transparente, seguro y rápido. Algunos de los problemas que podría solucionar el uso de Blockchain en los procesos electorales son los siguientes:

- Velocidad en el conteo: A comparación con la forma actual de conteo de votos el conteo de votos en blockchain sería visualizado de manera automática pues al tener los datos descentralizados se debe llevar todos los datos a un mismo lugar para contar si no que a medida que van creando los bloques se podría verificar el conteo de estos, siendo así
- Fraude electoral: Gracias al cifrado y descentralización de blockchain, y al hecho de que es incorruptible, cada registro es fácilmente verificable. Los votantes pueden registrar su voto sin revelar su identidad o preferencias políticas. Sabiendo que cada identificación puede atribuirse a un voto, no se pueden crear falsificaciones y la manipulación externa de éstos es imposible.

- Abstencionismo: Al proporcionar una manera irrefutable y fácil de votar desde un dispositivo, las cifras de abstencionismo probablemente disminuirían, debido a que

éste se convierte en un proceso rápido y sencillo, sin necesidad de movilizarse. Estas son algunas de las razones principales por las que muchas personas habilitadas para votar deciden no hacerlo, ya que se han presentado casos como el de Arizona en el 2016, donde los votantes debieron esperar hasta 5 horas para depositar su voto en una elección primaria. Incluso, la cantidad de votantes aumentarían debido a la solución que esta tecnología aportaría para la imposibilidad de desplazarse en zonas y regiones apartadas.

- Alteración de los votos realizados: Debido a que cada bloque tiene una referencia a el bloque anterior (el hash) y almacena los datos respecto a las votaciones, si posteriormente se quiere alterar alguno de estos bloques no se podría pues el Hash que se calcula para este bloque sería distinto al calculado previamente, es decir, el original que se encuentra en el siguiente bloque por lo que se podría dar cuenta que el bloque fue modificado.

5. Referencias

BBVA. (2019, septiembre 10). Cómo ‘blockchain’ puede cambiar la forma en que votamos. *BBVA NOTICIAS*. https://www.bbva.com/es/como-blockchain-puede-cambiar-la-forma-en-que-votamos/

Calcular hash Adler32 online. (s. f.). online-convert.com. Recuperado 28 de abril de 2021, de https://hash.online-convert.com/es/generador-adler32

El cifrado SHA-1 ya no es seguro: Google lo ha roto después de 22 años. (2017, febrero 23). ADSLZone. https://www.adslzone.net/2017/02/23/cifrado-sha-1-ya-no-seguro-google-lo-ha-roto-despues-22-anos/

Escritor, G. Á. M., & ElevenPaths, científico y conferenciante E. del Á. de I. y L. en. (2019, noviembre 12). *La criptografía insegura que deberías dejar de usar—Think Big Empresas*. Think Big. https://empresas.blogthinkbig.com/la-criptografia-insegura-que-deberias-dejar-de-usar/

Liebkind, J. (s. f.). *How Blockchain Technology Can Prevent Voter Fraud*. Investopedia. Recuperado 1 de mayo de 2021, de https://www.investopedia.com/news/how-blockchain-technology-can-prevent-voter-fraud/

Management, F. B. F. L. F. T. J. T. wrote about bitcoin for T. B. H. is a managing partner at D. C., Cryptocurrencies, T. A. of S. B. on, & Tatar, retirement R. T. B. editorial policies J. (s. f.). *Can Blockchain Technology Change How We Vote?* The Balance. Recuperado 1 de mayo de 2021, de https://www.thebalance.com/how-the-blockchain-will-change-how-we-vote-4012008

MD5: Vulnerabilidades y evoluciones (y II) - Think Big Empresas. (2013, noviembre 26). Think Big. https://empresas.blogthinkbig.com/md5-vulnerabilidades-y-evoluciones-y-ii/

¿Qué algoritmos criptográficos se deben emplear para cumplir con PCI DSS? (2016, septiembre 8). PCI Hispano. https://www.pcihispano.com/que-algoritmos-criptograficos-se-deben-emplear-para-cumplir-con-pci-dss/

¿Qué son los algoritmos de encriptación AES y 3DES? (2017, enero 26). *NÜO Planet: Sistemas de Control de Accesos y videointercomunicación digital IP*. https://nuoplanet.com/blog/algoritmos-encriptacion-aes-3des/

Secure Hash Algorithm. (2021). En *Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Secure_Hash_Algorithm&oldid=134915988

Uso de Blockchain: Lista de 20+ casos de uso de la tecnología Blockchain. (2019, julio 23). 101 Blockchains. https://101blockchains.com/es/uso-de-blockchain/

What Is Hashing? [Step-by-Step Guide-Under Hood Of Blockchain]. (s. f.). Recuperado 28 de abril de 2021, de https://blockgeeks.com/guides/what-is-hashing/