Power System Technology
ISSN:1000-3673

Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

A Two-Phase Data Integrity Verification using Cyclic Redundancy Check and SHA-256 Algorithm for Security of Cloud Data

Pradeep Semwal[1], Sanjay Sharma[2*], Nitin Uniyal[3]

psemwal2222@gmail.com[1], sanjaypokhriyal@yahoo.com[2*], nitinuniyalddn@gmail.com[3]
[1,2] School of Computer Application & IT, Shri Guru Ram Rai University,
Dehradun -248001, Uttarakhand, India.
[3]University of Petroleum & Energy Studies,
Dehradun -248001, Uttarakhand, India
*Corresponding Author

Abstract:

With the rising dependence on distributed computing, assuring the security of information put away in the cloud has become vital. Conventional strategies for information may not be adequate to handle the advancing digital dangers. This proposed paper proposes an enhanced efficient way to deal with upgrade cloud data security through a two-layered message trustworthiness check system that incorporates cyclic Redundancy Check (CRC) and hash capability strategies. By joining these two techniques, the proposed framework plans to give a strong cryptographic module for ensuring message integrity of data stored in the cloud . Hash capabilities are basic cryptographic apparatuses used to guarantee information uprightness in different applications. The proposed module uses CRC-8,CRC-16 and CRC-32 well tested divisor. The proposed method also uses SHA -256 hash Algorithm to calculate hash Code which is less prone to collision. The paper not only proposes strong message integrity module but additionally addresses the collisions and discuss thecomputation speed of proposed cryptographic module too.

Keywords: *Cloud data Security, Message Integrity, secure Devisor exchangeAlgorithm, Polynomial Division, Cyclic Redundancy Check, Hash function, SHA-256.*

Introduction:

Cloud Data [14] integrity is essential for information security, ensuring data accuracy, consistency, and non-alteration throughout its transmission over the channel. Error detection [4] means change in the received data which can be due to fluctuation in voltage level , thermal noise ,impulse noise [7] or due to tampering in data by unauthorized party. For computing point of view error detection should be at minimum cost [4] with respect to computing time [6], space and power consumption [18].

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

Cyclic Redundancy Check (CRC) [3] detects accidental or intentional changes in data during transmission or storage by generating a fixed-size CRC remainder a code word [20] which act as checksum. This checksum accompanies the data and, upon reception, the checksum or CRC remainder is recalculated. Comparing CRC values indicate data integrity, while differences signal potential corruption or tampering, necessitating retransmission or rejection of data block. CRC error detection method is based on binary division [7] which uses XOR [6] and shift [6] operation to calculate CRC remainder at sender end and its verification at receiver end. To compute an n bit binary CRC remainder, align the bits representing the input in a row and position the $(n+1)$ bit pattern representing the CRC divisor called as CRC polynomial [3] underneath the left row .CRC is capable of detecting single bit error and burst error of length equal to degree of CRC divisor i.e. n

if not longer than n bits, and the fraction of all longer error bursts that it will detect is approximately $(1 - 2^{-n})$.

- All burst errors of length less than equal to n.
- All burst errors affecting an odd number of bits.
- All burst errors of length equal to $n + 1$ with probability $\frac{2^{n-1}-1}{2}$.

CRC-8 is commonly used in low-speed communication systems, such as infrared communication ATM [24] and RFID systems. CRC-16 is used in a variety of communication systems, including conventional Ethernet [23] [24] with 100 Mbps bandwidth, Giga bit Ethernet [2] and HDLC. CRC-32 is used in many communication systems, including Ethernet, TCP/IP, and ZIP files. Research papers also reveals that CRC-32 is not secure and result in collisions for wireless sensor network [10] [11] WEP protection [22] , so hash function are preferred over it .

Hash functions[13] are mathematical algorithms transforming input data into fixed-size hash values, serving as unique digital fingerprints. They possess deterministic properties and pre-image resistance, making it computationally infeasible to reverse the process. Hash functions reliably detect alterations, deletions, or insertions within data sets, documents, or files, as even minor changes produce distinct hash important for data integrity across telecommunications, network protocols, file systems, databases, Cloud computing [14] and cryptography. Hash functions are integral to cryptographic techniques, digital signatures, and data integrity verification.

Since there were collision in CRC method so Hash function is used to further check the message integrity.A Hash Function $H(M)$ takes anbinary input M of arbitrary length and produces a fixed size Hash Value $H_i = H(M)$. The properties that define a secure Hash function include the following:

Deterministic Output: For the same input M,the Hash function $H(M)$ will always produce the same hash value H_i.

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

Pre-image Resistance: Given a Hash value H_i, it should be computationally infeasible to find a different input M such that $H(M)$=H_i

Second Pre-Image Resistance: Given an input M, it should be computationally infeasible to find a different inputs$M and\ M'$ such that $H(M) = H(M')$

Collision Resistance: It should be computationally infeasible to find two distinctM_1 and M_2 such that $H(M_{1)}$= $H(M_{2)}$

Collisions are incredibly unlikely. There are2^{256}possible hash values when using $SHA-256$, which makes it nearly impossible for two different documents to coincidentally have the exact same hash value. Probability of two texts producing the same hash value is incredibly small, approximately2^{-256}. It means one collision out of
$115,792,089,237,316\ ,195\ ,423,570,985,008,687,907,853,269,984,665,640,564,039,457$

$,584,007,913,129,639,935$ total hash values is approximately $1.1579208923732 \times 10^{77}$.

Algorithm like SHA-256 and SHA-512 are used to calculate the unique finger print of message called Hash code. Various Research paper reveals that algorithm to computer hash code like SHA-256 [18] when implemented on 139.04 MHz processor(very less configuration) with bandwidth of 1.04 Gbps it consumes .072 watt power [18]. Also when SHA-256 implemented in Intel 14nm technology can improve three times with 50.7 % power reduction [15].Also the strength of SHA algorithm also depends on no of rounds [16] it uses.

CRC and hash functions are indispensable for safeguarding data integrity, enabling organizations to uphold trust, reliability and accuracy in digital operations and communications.

Proposed Cryptographic Module:

Let us assume a Message M of arbitrary size to be transmitted from one end to another i.e. Sender (S) to Receiver(R) .

Pre computation step:

Step 1. Convert Message M into i binary blocks say D_1 to D_i.

Criteria for blocks sizes: Minimum block size is 64 bytes and maximum block size is 64 KB.Moderate Block size is calculated using code in Python shown below

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

 # user defined function made in python to calculate block size

```
defget_block_size(file_size):
    min_size=64                  # minimum file size 0.1KB
    max_size = 64 * 1024         #maximum file size  64 KB
    size=file_size//20000
    if size  <min_size:
    size=min_size
  elif    size >max_size:
  size=max_size
    return size
```

Step 4: Choose CRC Devisor

Proposed cryptographic module uses three different unique divisors depending on the block size . Any one divisor is chosen by proposed cryptographic module , choosing divisor for different is to reduces the **collisions.** The more will be the size of divisor less will be the chances of collision. The relationship is as follows

Size of divisor $\propto$ *size of data block* $\propto$ *size of file*

CRC Devisor	*Polynomial*	*Polynomial (in Hex)*	*Polynomial(ininteger)*
$CRC-8$ (for small size file)	x^8+x^2+x+1	$0x107$	263
$CRC-16$ (for moderate size file)	$x^{16}+x^{13}+x^{12}+x^{10}+x^9+x^7x^6+x+1$	$0x136C3$	79555
$CRC-32$ (for large size file)	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$	$0x104C11DB7$	4374732215

Table 1 . CRC polynomial used by proposed cryptographic module

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

Step4. Exchange Divisor using Secure Exchange Devisor Algorithm

Secure Exchange Devisor Algorithmuses number theory.Two authenticated parties i.e. the sender(S) and receiver(R) agree on two numbers p and g. where p is a prime number and g is its primitive root.Primitive root g has property that if p is prime number than $g^i \text{mod } p$ will always produce unique value and no duplicate value for all values of i .

steps	Sender(S) End	Receiver (R)End
1	p and g are agreed on public numbers or public keys by both parties	p and g are agreed on public numbers or public keys by both parties
2	x is a private key or random number which is secretly selected.	y is a private key or random number that is secretly selected.
3	Equation to generate the numbers $n_1 = g^x \ mod \ p$	Equation to generate the numbers $n_2 = g^y \ mod \ p$
4	After the exchange of numbers S receives n_2	After the exchange of numbers R receives n_1
5	S generates a secret divisor $G_1(x)$ by using the received number n_2 $G_1(x) = n_2^x \ modp$	R generates a secret divisor $G_2(x)$by using the received number n_1 $G_2(x) = n_1^y modp$
6	$G_1(x) = G_2(x)$, it means both ends knows the common secret devisor say $G(x)$	

Table 2 .Computation and secure exchange of Divisor

This is how the sender(S) and receiver(R) securely exchange divisor $G(x)$with each other without sharing it, actually they exchange two numbers $n_1$and n_2.This is secretly exchanged divisor agreed polynomial between both the parties.

Cryptographic Module at Sender end:

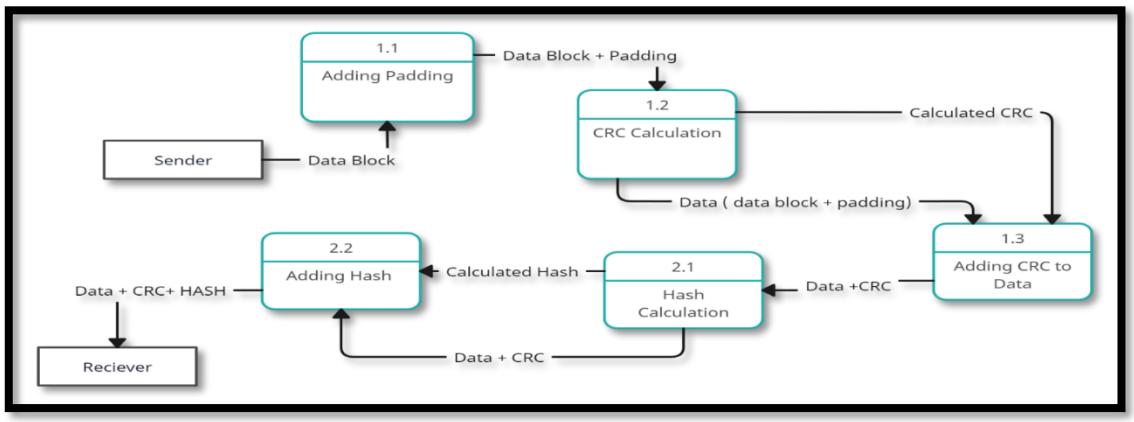


Figure 1: Graphical representation of steps used by cryptographic module at sender end.

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

CRC operates on binary data block comprising of thebinary coefficients of a unique polynomial of degree n.

A data block D can be represented mathematically as a unique polynomial of degree n in variable x as below:

$$D(x) = d_n x^n + d_{n-1}x^{n-1} + \ldots\ldots + d_1 x + d_0$$

where eachd_i isa binary coefficient such that $d_i \in \{0,1\}\ \forall\ 0 \leq i \leq n$and x represents the polynomial variable, Similarly, the agreed upon and securely exchanged CRC polynomial $G(x)$of degree $k \leq n$ is represented as

$$G(x) = g_k x^k + g_{k-1}x^{k-1} + g_{k-2}\,x^{k-2} + \cdots g_1\,x + 1\text{ , where } g_i \in \{0,1\}\ \forall\ 1 \leq i \leq n$$

The CRC (Cyclic Redundancy Check) division process can be mathematically modelledusing the polynomial long division.

Step 1: Padding

Append $k-1$ zeros to the end of the data sequence D so that$P(x) = D(x).x^{k-1}$ is a polynomial of degree $n+k-1$, where k is the degree of divisor$G(x)$.

Step 2: Division of padded data polynomial$P(x)$with Securely Exchanged Divisor $G(x)$

Divide the polynomial $P(x)$ by the CRC polynomial $G(x)$ using polynomial long division. The result is the quotient $Q(x)$ and CRC remainder $R(x)$ of degree $k-1$ such that

$$P(x) = Q(x).G(x) + R(x)$$

Step 3. Removal of padded zeros from the data and padding of CRC Remainder in place of it

CRC Remainder $R(x)$thus obtained from the division is substituted in place of the padded zeros of the original data so that $D(x).R(x)$ is the resulting polynomial for next step.

Step 4. Computation of unique Hash Code H_1 using SHA -256 Algorithms

Step5. Append Hash code H_1 with $D(x).R(x)$

Finally,$D(x).R(x).H_1$ is transmit over the cloud

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

Cryptographic Module at Receiver End:

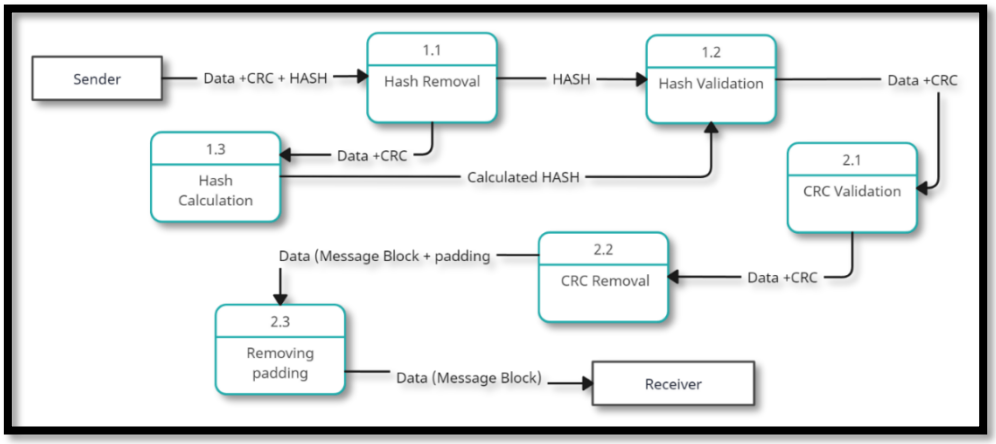


Figure 2: Graphical representation of steps used by cryptographic module at Receiver end.

Step 1. Remove H_1 from $D(x).R(x).H_1$ store it separately for future comparison.

Step 2. Receiver again divides $D(x).R(x)$with securely exchanged divisor $G(x)$ to obtain the remainder$R'(x)$ of degree $k-1$

$$D(x).R(x) = G(x).Q'(x) + R'(x)$$

where $Q'(x)$ is some quotient polynomial.

If all the coefficients of this newly calculated remainder $R'(x)$are zerothen this suggests no data tampering is done during transmission.Data received is exactly the same as it was sent and thus message integrity is not violated.

One level of message integrity is checked and verified at the receiver end.Now the method further proceeds for second level check.

Step 3. Calculate unique hash Code H_2at the receiver end using $SHA-256$ algorithm.

Step 4. If $H_1 = H_2$ it means no data tampering during the transmission.Data sequence corresponding to the$D(x)$thus received is exactly same as it was sent.

Illustrations:

Table 3. Illustrates how CRC Divisor i.e 9 is securely exchanged between two authenticated parties where actually Divisor is not exchanged only two numbers n_1 and n_2 are exchanged which are computed based on value of p, g, x, y. In our proposed cryptographic module 263

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

(CRC -8 divisor) or 79555 (CRC-16 divisor) or 4374732215 (CRC-32 divisor) will be securely exchanged. Small number is intensely chosen for illustration purpose only to make computation easy and understandable.

Table 4. Illustrates that SHA-256 always produces unique hash code and proves to be very strong with almost zero collisions

Steps	**Sender end**	**Receiver End**
1	$p = 11$ and $g = 7$	$p = 11$ and $g = 7$
2	$x = 3$ random number secretly selected	y =6 random number secretly selected
3	Compute $n_1 = 7^3 mod\ 11 =$ 343%11=2	Compute $n_2 = 7^6 mod\ 11 = 117649\%11=4$
4	Send n_1 to receiver	Send n_2 to sender
5	Compute Divisor $4^3\ mod\ 11 = 9$	Compute Divisor $2^6\ mod\ 11 = 9$

Table 3. Computation of Securely exchanged Divisor

Input Message	*Hash code (in Hex value)*
Hello World	a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e
Hello world	64ec88ca00b268e5ba1a35678a1b5316d212f4f366b2477232534a8aeca37f3c
hello World	db4067cec62c58bf8b2f8982071e77c082da9e00924bf3631f3b024fa54e7d7e
Hello word	aea1d1146a00e1c55e49c7837c224ecfb76ca0337fd4bb6dc09e892ca0190119
Hello worlde	a238f59ce202ae7ebfd6bc5da6b5540e1fab7ffc49f311e9d055d66cdd230065
Hello word	aea1d1146a00e1c55e49c7837c224ecfb76ca0337fd4bb6dc09e892ca0190119
hello Word	65d86dd317089a03da204a364d2fcd2f18157faccc56c81b7b94ebfb588163d5
hello word	f0da559ea59ced68b4d657496bee9753c0447d70702af1a351c7577226d97723
hell World	3dc3a752f0275553d709656a0a83344acfee9b5cec9784576ac3f54bb499f4c8
hell world	2fe4d4a5963f28b77737c091c436096beee0b74fabb9fcdcd2a4d8859d2099a3

Table 4. Unique Hash code generated by SHA-256

Power System Technology
ISSN:1000-3673

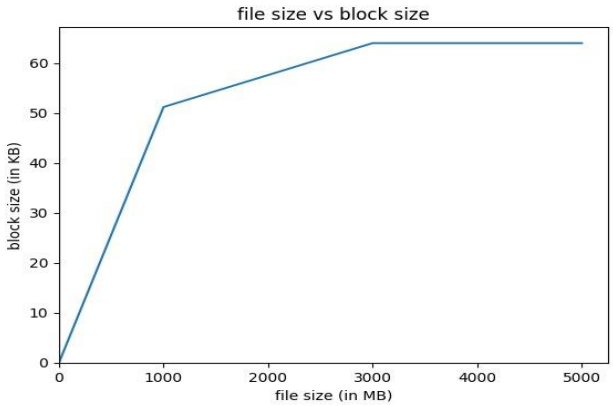
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

Result Analysis: The computation time for different file sizes and block sizes with different CRC Divisors along with SHA-256 are tabulated and shown below for analysis purpose. The proposed cryptographic module proves to be stable and reliable for better message integrity of transmitted message.

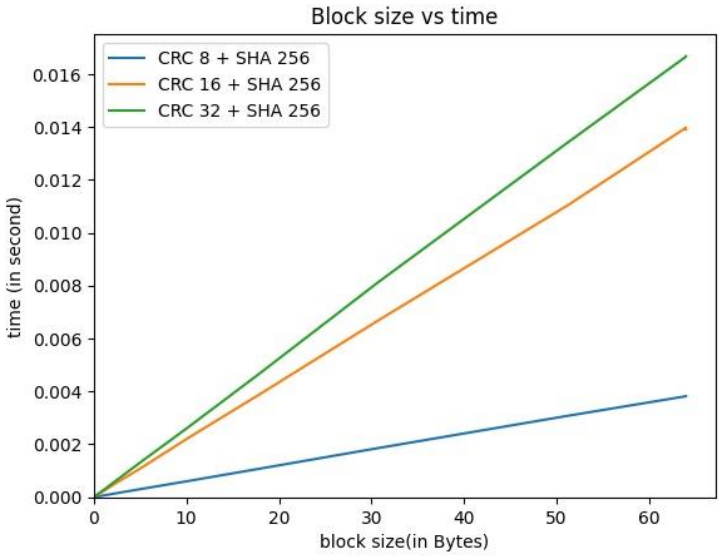
File Size (in bytes)	*File Size (in MB)*	*Block Size (bytes)*	*Number of Blocks*	*Proposed CRC + SHA-256*	*Time per Block (sec)*	*Total time to process file (sec)*
1048576	1	64	16645	CRC8	0.0000025	0.0461
1048576	1	64	16645	CRC16	0.0000026	0.04221
1048576	1	64	16645	CRC32	0.0000028	0.04301
2621440	2.5	131	20165	CRC8	0.0000028	0.0571
2621440	2.5	131	20165	CRC16	0.0000028	0.05701
2621440	2.5	131	20165	CRC32	0.0000035	0.05599
5242880	5	262	20088	CRC8	0.0000036	0.071
5242880	5	262	20088	CRC16	0.0000038	0.073
5242880	5	262	20088	CRC32	0.0000051	0.0761
10485760	10	524	20050	CRC8	0.0000057	0.10233
10485760	10	524	20050	CRC16	0.0000053	0.11509
10485760	10	524	20050	CRC32	0.0000093	0.1072
26214400	25	1310	20027	CRC8	0.0000099	0.18632
26214400	25	1310	20027	CRC16	0.0000094	0.19845
26214400	25	1310	20027	CRC32	0.0000166	0.18725
52428800	50	2621	20011	CRC8	0.0000193	0.3328
52428800	50	2621	20011	CRC16	0.0000179	0.38638
52428800	50	2621	20011	CRC32	0.0000296	0.35895
104857600	100	5242	20008	CRC8	0.0000346	0.59292
104857600	100	5242	20008	CRC16	0.0000323	0.6931
104857600	100	5242	20008	CRC32	0.0000547	0.64535
209715200	200	10485	20004	CRC8	0.0000661	1.09424
209715200	200	10485	20004	CRC16	0.0000617	1.32258
209715200	200	10485	20004	CRC32	0.0000921	1.23393
367001600	350	18350	20002	CRC8	0.0001136	1.84293
367001600	350	18350	20002	CRC16	0.0001057	2.27145
367001600	350	18350	20002	CRC32	0.000159	2.11385
629145600	600	31457	20001	CRC8	0.000187	3.18064
629145600	600	31457	20001	CRC16	0.0001709	3.73942
629145600	600	31457	20001	CRC32	0.0002606	3.41731
1048576000	1000	52428	20001	CRC8	0.0003121	5.21138

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

1048576000	1000	52428	20001	CRC16	0.0003235	6.24162
1048576000	1000	52428	20001	CRC32	0.0004271	6.47035
3145728000	3000	*65536*	48001	CRC8	0.000527	20.50008
3145728000	3000	*65536*	48001	CRC16	0.0004742	25.29438
3145728000	3000	*65536*	48001	CRC32	0.0003331	22.76073
5242880000	5000	*65536*	80002	CRC8	0.0004084	26.64791
5242880000	5000	*65536*	80002	CRC16	0.0003849	32.67216
5242880000	5000	*65536*	80002	CRC32	0.0000025	30.79589

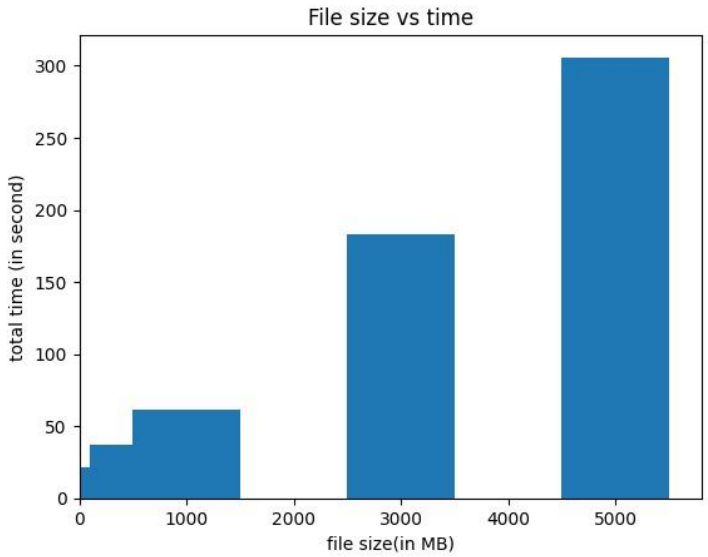


Graph 1: Block size vs File Size

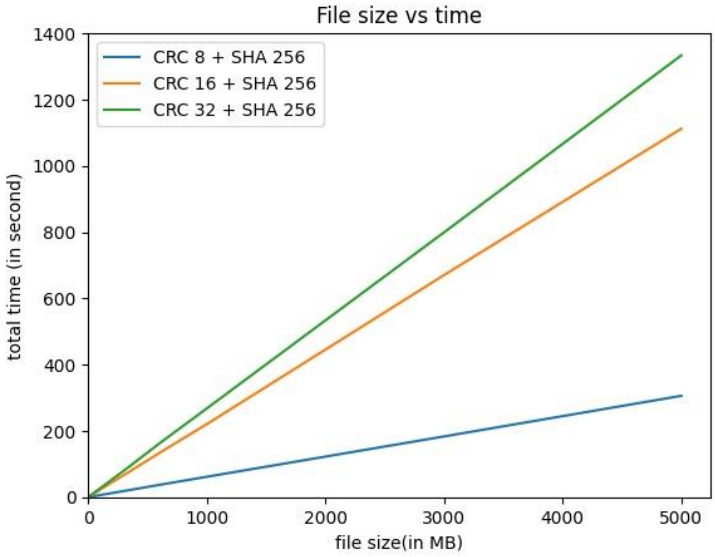


Graph 2. Computation time vs Block Size

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024



Graph 3. Computation time vs File Size



: *Graph 4. Computation time vs File Size for different CRC Divisor with SHA-256*

Conclusion:

1. CRC (Cyclic Redundancy Check) method which uses polynomial division method, can detect single bit error and burst error also, but if the length of the burst becomes more than the degree of divisor polynomial then CRC method is not useful and results in collision. Mathematically, if τ is the set of $d_i's$ which gets corrupted such that the cardinality of τ is not more than k then the method succeeds otherwise; the collisions

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

may occur and method suffers. Whilst the proposed method addresses this using Hash function in succession.

2. Generally, the block size is directionally proportional to size of file, but later the block size calculated by the proposed cryptographic module eventually becomes a constant even if the file size increases. It stabilizes the proposed method and improves its overall performance.
3. For different sizes of CRC divisors viz. CRC-8, CRC-16, CRC-32 with SHA-256 the computation time of the proposed cryptographic module is different .Also computation time for CRC-32 + SHA-256 is less as compared to CRC-16 + SHA-256, So CRC-32 + SHA-256 can be preferred over it with respect to computation speed and collisions for better data security and verification of message integrity

References:

1. Sharma, D. D., & Choudhary, S. (2023, August). Pipelined and Partitionable Forward Error Correction and Cyclic Redundancy Check Circuitry Implementation for PCI Express® 6.0. In *2023 IEEE Symposium on High-Performance Interconnects (HOTI)* (pp. 1-8). IEEE.
2. Kishore, P., Pal, B. A., Kishore, L. N., & Revathi, C. V. (2023, May). Implementation of Table-Based Cyclic Redundancy Check (CRC-32) for Gigabit Ethernet Applications. In *2023 4th International Conference for Emerging Technology (INCET)* (pp. 1-4). IEEE.
3. Barton, D. (2022). *A Flexible Low Power Cyclic Redundancy Check Algorithm Using Reduced Lookup Tables in Parallel* (Doctoral dissertation, The University of Texas at San Antonio).
4. Xiaoqing, G., Guangzu, L., Jun, Z., & Linlin, S. (2022, August). Compiled code study of joint parity and cyclic redundancy check. In *2022 IEEE 5th International Conference on Electronic Information and Communication Technology (ICEICT)* (pp. 77-80). IEEE.
5. Kim, J., Kwon, S., Noh, J., & Shin, D. J. (2022). Construction of cyclic redundancy check codes for SDDC decoding in DRAM systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *70*(2), 736-740.
6. Chi, M., He, D., & Liu, J. (2018, November). Fast software-based table-less algorithm for CRC generation. In *2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC)* (pp. 544-549). IEEE.
7. Uniyal, N., Dobhal, G., Rawat, A., & Sikander, A. (2021). A novel encryption approach based on vigenere cipher for secure data communication. *Wireless Personal Communications*, *119*, 1577-1587.
8. Jain, S., & Chouhan, S. S. (2014). Cyclic Redundancy Codes: Study and Implementation. *International Journal of Emerging Technology and Advanced Engineering (IJETAE)*, *4*, 213-217.
9. Zhang, Y., Luo, X., Zhu, X., Li, Z., & Bors, A. G. (2020). Enhancing reliability and efficiency for real-time robust adaptive steganography using cyclic redundancy check codes. *Journal of Real-Time Image Processing*, *17*(1), 115-123.
10. Koopman, P., & Chakravarty, T. (2004, June). Cyclic redundancy code (CRC)

Power System Technology
ISSN:1000-3673
Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

polynomial selection for embedded networks. In *International Conference on Dependable Systems and Networks, 2004* (pp. 145-154). IEEE.
11. Ezea, M. O., Osuagwu, H. O., & Ahaneku, M. A. (2020). Performance analysis of cyclic redundancy check (CRC) error detection technique in the wireless sensor network. *Int. Res. J. Eng. Technol*, *7*(6), 4104-4110.
12. Engdahl, J. R., & Chung, D. (2014, October). Fast parallel CRC implementation in software. In *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)* (pp. 546-550). IEEE.
13. Ali, A. A. M. A., Hazar, M. J., Mabrouk, M., & Zrigui, M. (2023). Proposal of a Modified Hash Algorithm to Increase Blockchain Security. *Procedia Computer Science*, *225*, 3265-3275.
14. Semwal, P., Sharma, S., & Uniyal, N. (2022). A Three Level Cryptographic Security Module To Ensure Security Of Health Care Data Stored In The Cloud. *NeuroQuantology*, *20*(14), 1420.
15. Wu, R., Zhang, X., Wang, M., & Wang, L. (2020, February). A high-performance parallel hardware architecture of SHA-256 hash in ASIC. In *2020 22nd International Conference on Advanced Communication Technology (ICACT)* (pp. 1242-1247). IEEE.
16. Biham, E., & Chen, R. (2004). Near-collisions of SHA-0. In *Advances in Cryptology–CRYPTO 2004: 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004. Proceedings 24* (pp. 290-305). Springer Berlin Heidelberg.
17. Semwal, P., Sharma, S., & Uniyal, N. (2023). An Efficient Subkeys Generation Algorithm For Encryption Of Cloud Data. *SJIS-P*, *35*(1), 476-482.
18. Kieu-Do-Nguyen, B., Hoang, T. T., Pham, C. K., & Pham-Quoc, C. (2021, October). A Power-efficient Implementation of SHA-256 Hash Function for Embedded Applications. In *2021 International Conference on Advanced Technologies for Communications (ATC)* (pp. 39-44). IEEE.
19. F. Chabaud and A. Joux, *Differential Collisions in SHA-0, in Advances in Cryptology - CRYPTO'98, LNCS 1462, pages 56-71, Springer-Verlag, 1998.*
20. Singh, A. K. (2017, April). Comprehensive study of error detection by cyclic redundancy check. In *2017 2nd International Conference for Convergence in Technology (I2CT)* (pp. 556-558). IEEE.
21. Klimenko, S. V., Yakovlev, V. V., & Sokolov, B. V. (2018). The Study of Implementations of CRCs Algorithms. In *Workshop Computer Science and Engineering in the framework of the 5th International Scientific-Methodical Conference «Problems of Mathematical and Natural-Scientific Training in Engineering Education»* (No. 1, pp. 27-32)
22. Grover, A., & Singh, S. (2015). Comparative Analysis of CRC-32 and SHA-1 Algorithms in WEP. *Advanced Engineering Technology and Application*, *4*(1), 5-10.
23. Zolfaghari, B., Sheidaeian, H., &Mozafari, S. P. (2011, June). Systematic selection of CRC generator polynomials to detect double bit errors in Ethernet networks. In *International Conference on Computer Networks and Communications* (pp. 228-235). Berlin, Heidelberg:

Power System Technology
ISSN:1000-3673

Received: 16-01-2024 Revised: 12-02-2024 Accepted: 07-03-2024

24. Hajare, P. S., & Mankar, K. Design and Implementation of Parallel CRC Generation for High Speed Application. *IOSR J. VLSI Signal Process.(IOSR-JVSP, vol. 5, no. 3, p. 1, 2015, doi: 10.9790/4200-05320105*.
25. Sahu, A., & Ghosh, S. M. (2017). Review paper on secure hash algorithm with its variants. *International Journal of Technical Innovation in Modern Engineering & Science*, *3*(05).
26. Chaves, R., Sousa, L., Sklavos, N., Fournaris, A. P., Kalogeridou, G., Kitsos, P., & Sheikh, F. (2016). Secure hashing: Sha-1, sha-2, and sha-3. *Circuits and systems for security and privacy*, 105-132.
27. Uniyal, N., Dobhal, G., & Kothiyal, A. D. (2021). An Improvement in Key Domain Maximization Technique by Entropy Maximization. In *Proceedings of Integrated Intelligence Enable Networks and Computing: IIENC 2020* (pp. 681-687). Springer Singapore.
28. Eastlake 3rd, D., & Hansen, T. (2006). *US secure hash algorithms (SHA and HMAC-SHA)* (No. rfc4634).
29. Algredo-Badillo, I., Feregrino-Uribe, C., Cumplido, R., & Morales-Sandoval, M. (2013). FPGA-based implementation alternatives for the inner loop of the Secure Hash Algorithm SHA-256. *Microprocessors and Microsystems*, *37*(6-7), 750-757.
30. Van Tilborg, H. C., & Jajodia, S. (Eds.). (2014). *Encyclopedia of cryptography and security*. Springer Science & Business Media.
31. Ritter, T. (1986). The great CRC mystery. *Dr. Dobb's Journal*, *11*(2), 26-34.
32. Van Tilborg, H. C., & Jajodia, S. (Eds.). (2014). *Encyclopedia of cryptography and security*. Springer Science & Business Media.
33. FIT BUT, B., & FI MU, B. Formal Verification of the CRC Algorithm Properties.
34. National Institute of Standards, Technology (US), & Computer Systems Laboratory (US). (1994). *Security requirements for cryptographic modules* (Vol. 140, No. 1). Computer Systems Laboratory, National Institute of Standards and Technology.