CrAlSim: A Cryptography Algorithm Simulator

Avinash Singh[1], Swapnil Mane[2], Mihir Bist[3], Samuel Jacob[4]

[1]*B.E. Information Technology, Dept. of Information Technology, Vidyalankar Institute of Technology, Maharashtra, India*
[2] *B.E. Information Technology, Dept. of Information Technology, Vidyalankar Institute of Technology, Maharashtra,India*
[3]*B.E. Information Technology, Dept. of Information Technology, Vidyalankar Institute of Technology, Maharashtra, India*
[4]*Professor, Dept. of InformationTechnology, Vidyalankar Institute of Technology, Maharashtra, India*

---***---

Abstract- *Encryption algorithms are an essential part of Information Security, as they safeguard the messages sent between users, and also protect the integrity and confidentiality of the system. For students to understand the internal working of complex algorithms, it's requisite that they don't mug up the steps happening in the cipher, but rather will learn better if it happens to be a graphical visualization of every event happening in the algorithm. This paper describes working of cryptographic algorithm simulation system. CrAlSim is based on matrix formation, stepwise color-based value change in code of the crypto algorithms, highlighted by each function showing the calculation right from the first to last step of encryption. The system created highlights how based on user inputs for a plain message or a key wherever required, how the actual inside conversions happen in an algorithm which led to the encryptions, be it a simple monoalphabetic cipher like Affine or block cipher like AES (Advance Encryption Standard). The encryption functions adjusted with the simulations are programmed in JavaScript, and simply HTML and CSS are used to display the web elements and styling, so understanding the source code even for creating of other algorithm visualization systems is comprehensible and apprehensible for a normal user. This paper elaborates the mechanism of CrAlSim and discusses the results obtained using it.*

***Key Words*: Simulator; Encryption; Decryption; AES; Affine; RSA; Peer to peer message confidentiality; Algorithm simulation.**

1. INTRODUCTION

Nowadays, students often face the dilemma of choosing whether to take up security as one of their core domains. Since most students seem to face the problem in understanding various ciphers, they never chose to look further on how security works more than just simple cipher exchanging acknowledgement requests in SSL of presentation layer [1]. So, every IT student needs an easier guide to comprehend ciphers, be it of various types in an easy format. Since a picture is worth a thousand words, a step visualization simulation for demonstrating a cipher working with all the components running over every input character is much better than 500 pages of algorithm in theory mode of a published author. As most students face this issue after reaching their second year in IT or Computer branch, they are the target audience of this website. Encryption algorithms are very essential to be learnt in the current era, as malicious users most often are looking for unsecured data streams over the internet. If any attacks are launched, it can prove to be hazardous if sensitive data is revealed, and Government information, data related to financial sectors and other critical data can be leaked to foreign parties if it isn't encrypted properly. Hence these encryption algorithms need to be studied, and can be better studied by trainee engineers or IT students if they learn it via a visualization tool which will enhance their perception of all stages happening in the code of certain algorithms [9]. Such a tool is CrAlSim which provides a simulator for essential cryptographic algorithms.

1.1 Simulation Process

Visualization of algorithms is done using simple functions which target over the events which can be made graphical as so students or any user perceives those events in motion pictures or color highlights and hence better representation of various operational fields of code can be shown. First step is selection of cipher as in which category of the algorithm needs to be visualized. For a single word string, monoalphabetic ciphers such as Affine are useful as their encryption can be based on just a certain length and for a block of letters; AES (Advanced Encryption Standard) cipher can be chosen. User then needs to input the proper type of the plain text message, as certain criteria is set for certain ciphers based on length of string, or it's standard language used. Then the key is selected, or generated as per conditions, as RSA needs very high values for proper security. Encryption and decryption buttons should be used to generate cipher text or decipher plain message. Note must be made of the special instructions in the process, explaining as in which kind of key to select, or what conversions are happening. Then controlling the flow of simulation using the start, play, pause or reset buttons.

2. LITERATURE SURVEY

Through this section, we summarize some of the existing research work on either encryption or general simulation systems.

In January 2017, U. Arom-oon had proposed an AES cryptosystem for a small-scale network [2]. For low power sensor devices, the scale of the network based on wireless communication is quite small. To protect electronic data, FIPS 197 standard [12] encryption is used. Say a UAVs wireless communication network has microcontroller which works on real time operating system, so the main basis of security is chosen from the code book which is one mode of AES method. In real time scenarios, scheduler decides which role to get the nod based on the event having the top priority. Hence no clash of system calls happen in the system. The basis of the accomplishment is decided by the quality of security in communication of UAVs which consist of control and telemetry commands. The target hardware is implemented on the arm cortex-M4.

In December 2020, C. R. Martin proposed a system of A New Simulation Algorithm for PDEVS (Discrete Event Systems Specification) Models [5]. The algorithm deals with the timing of code executions, as functions can't be controlled in their flow of system calls just to wait for an animation packet to be displayed. In case of simulations, the user deals with the working of only general models and the corresponding DEVS simulations occur when the scheduled events are triggered. When system call occurs simultaneously inside a function for different events, processing happens with event A being executed firstly and later only event B is executed. But if animation call asks for the inputs simultaneously for different events, it may lead to incorrect and non-requisite results for simulation. So, in this paper, proposal of a new algorithm is laid out which makes sure that the model is such that output section of result is sent only on the condition that it's given to a corresponding event at the appropriate simulation time received by the model.

In November 2021, A. Kabulov and M. Berdimurodov proposed a system of optimal representation in the form of logical functions of microinstructions of cryptographic algorithms (RSA, El-Gamal) [6]. According to the present study, the algorithms can be described by a set of micro-instructions consisting of a sequence of operators. Based on an analysis of graphs, logic, algebraic, and matrix representations of algorithms, an implementation minimal logical representation for hard-drive implementation is constructed.

In September 2020, RahmaIsnaini Masya; Rizal Fathoni Aji; Setiadi Yazid proposed the Comparison of Vigenere Cipher and Affine Cipher in Three-pass Protocol for Securing Image [7]. The aim of this study was to utilize Vigenere Cipher and Affine Cipher in Three-pass Protocol and compare their encryption results and execution time at each stage of the protocol. The findings indicate that Affine Cipher outperforms Vigenere Cipher in terms of encryption results within the Three-pass Protocol. Different types of files, such as text, binary encoding for images, audio, and video, are frequently exchanged through the internet. The study focuses on securing images using classical cryptography algorithms. Vigenere Cipher and Affine Cipher are implemented in Three-pass Protocol to eliminate the need for key exchange. The key is kept secure by each sender and recipient, making it impossible for attackers to obtain it. But, the execution time for Affine Cipher in Three-pass Protocol is longer than that of Vigenere Cipher.

In August 2014, Y. Li, H. Zhang, H. Ju and X. Jiang proposed a Visual Simulation Technique of Decision Making of Interactive Stand Management Methods [10]. To adapt to intensive forest management, a technique platform was created that employs multiple visual simulation techniques. Thirteen stand management-related activities were customized and presented graphically using the WF technique. A flow model for decision-making during interactive stand management was developed based on human-computer interaction. The GDI+ drawing technique was used to simulate statistics of stand structure and state in 2D, while the MOGRE technique was used to simulate the stand scene in 3D. The effectiveness of the method was demonstrated by carrying out a case study of accretion cutting in a Chinese fir plantation, which showed that the method was flexible, visible and operable, and could be used to create stand management flow models., The aforementioned approach has the ability to replicate the decision-making process involved in selecting the most suitable method from a set of techniques based on the decision-making criteria. It can be effectively implemented in stand management practices and can optimize the method selection by taking into account the specific stand management objectives, ultimately enhancing the overall scientific management proficiency.

3. PROPOSED SYSTEM

With a view to simulate various cryptographic algorithms, there arises the feeling of obligation to create functions based on stepwise handling of the message data. This occurs when various functions are written as per the flow of the code, synchronized with each cipher generation, setting color divisions, creating loops for multiple steps and conditions. After simulation is complete, final result is shown by displaying the resultant string, by rather selecting the comparison of the plain text, or in different format based on user selected cipher. The simulator [11] parses all the values of calculations in a table which show the inside configurations happening in each step, be it alphabets or numbers fed into a formula, or passing through rounds of a code.

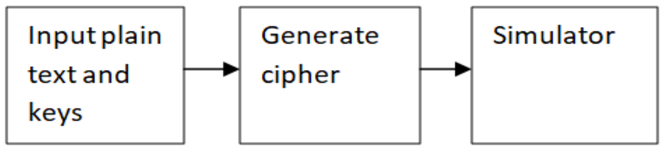


Fig 3.1: Block diagram of CrAlSim

Detailed explanation about modules in Fig 3.1 is given below-

1. Input plain text and keys module: In module one, once the user selects the type of cipher he needs, he must input his plain text message, which can be either in standard format, or can include basic punctuation in conjunction with numbers. The key or keys to be selected are mentioned in the note, for the criteria needs to be satisfied over which types of keys can be selected, e.g., for Affine cipher two keys are to be selected and they should be co-prime, in AES block cipher, a key can be 128, 192, or 256 bits. For our project permissible value chosen is 128 bits only, i.e., 16 characters.

2. Generate cipher module: In module two, we choose the encryption option. By giving in proper inputs for plain text message and keys, we can generate a cipher by clicking on the encrypt button. Following output is also shown on a general scale as in for Affine cipher, transformation of all alphabets is shown, hence user can tally how the change happened in original message and encrypted keyword.

3. Simulator: It consists of a JS code which checks all the necessary conditions for encryption to begin, and then creates the graphical representation of the working of the code, based on each step, value is fetched into matrices or tables, each new alphabet or number currently under operation is highlighted by a color or style which indicates the current step in progression, final transformations are duly saved in a all calculated values table so user can go through all the formula and numeric changes happened over each step or round.

4. IMPLEMENTATION AND RESULTS

4.1 Implementation

As process is initiated in the first module, encryption process occurs. User goes to homepage of our website and selects the cryptographic algorithm he wants to work upon, be it affine cipher of monoalphabetic type, AES of the category included in block cipher or RSA [13] a part of asymmetric cipher. Then the user enters his plain text message, can be in alphabetical form of characters, punctuation marks or numbers, then an appropriate key is selected. For affine cipher, key A and B should be co-prime of modulo m i.e., GCD of key and modulo m should be 1 as other combination result in two alphabets having same character as a cipher, so the inverse condition fails. Finally when all the clauses are satisfied, user can click on encryption button and generate the cipher.

In the second module, a graphical user interface (GUI) is rendered, as this is a tab where simulator works on the simulation [4], or graphical visualization of internal working of the algorithms. In correspondence to the highlighting of changes occurring in the code with respect to plain text message, internal cipher code function runs in synchronized manner. Based on that, value transformations happen which are exhibited via tables, matrices, or moving of object values. Each round of the function is demonstrated for each block of code or a single character, and to control the narrative over whether to watch all the steps or just comprehend one of the transformations and then skip to the final output, set controls are provided such as play, pause, skip and start buttons. Internal working of all calculations is shown in a table, or as in AES simulation, All Value button functionality is provided where all XOR, multiplication operations are accounted for each round. Final output value is converted back into a string, as calculations inside the function happen for hexadecimal values. So, the resultant cipher is returned back to the user in module one [15].

Flow chart for CrAlSim system is shown below:

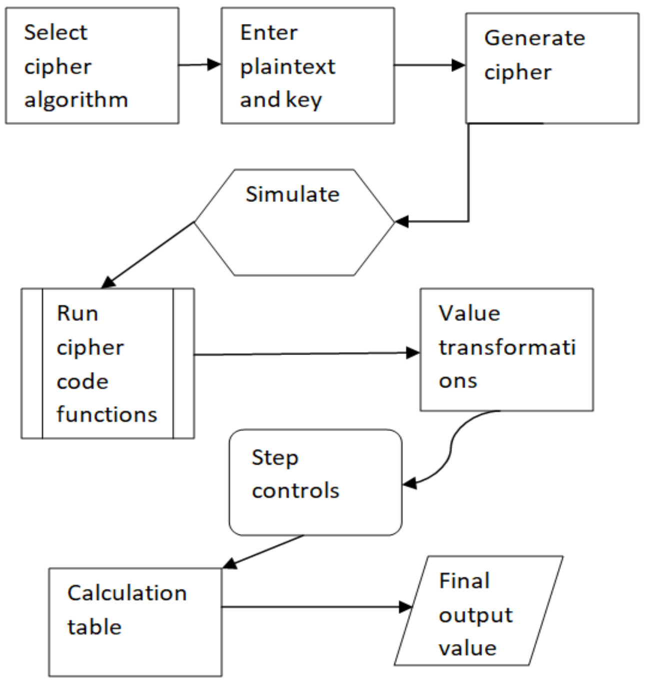


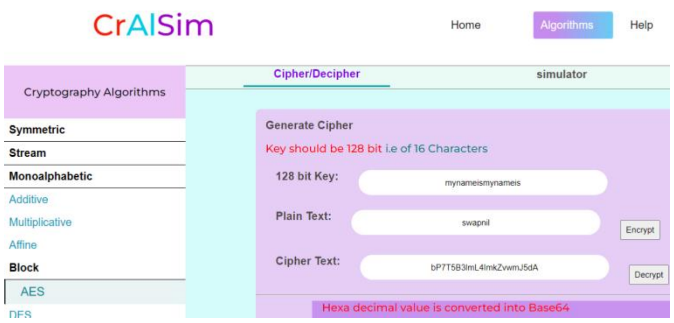
Fig 4.1: Flow chart for working of CrAlSim

As shown in Fig4.1, CrAlSim comprises of two modules, one displaying the basic encryption and other the actual simulation of internal working of algorithm.

4.2 Results

Examples when checked through the system showed accurate visualization graphics as intended and suited for the cryptographic algorithms.

AES:



Key should be 128 bit i.e of 16 Characters
128 bit Key: mynameismynameis
Plain Text: swapnil
Cipher Text: bP7T5B3lmL4ImkZvwmJ5dA
Hexa decimal value is converted into Base64

Fig 4.2.1: AES MAIN PAGE

For plain text 'swapnil' and using 128 bits key 'mynameismynameis', cipher text generated after encryption was 'bP7T5B3lmL4ImkZvwmJ5dA'.

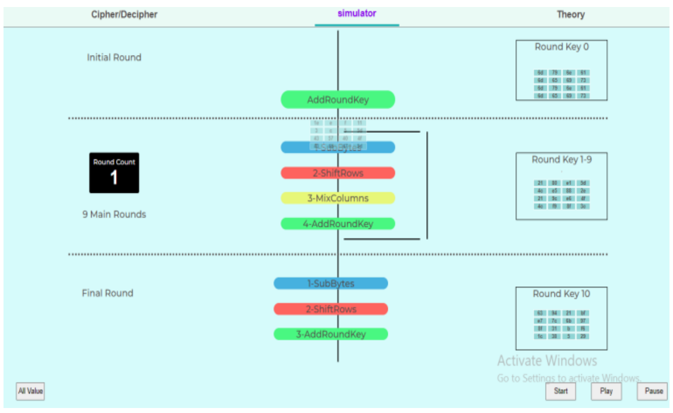


Fig 4.2.2: Initial round value pass

In initial round only XOR operation is performed between plain text converted into hexadecimal format put in a 4*4 array, (just like a matrix is created in hill cipher where each letter is represented by a number modulo 26) [8] along with the event round key word [0..3], and the output is calculated to the repetitive steps later in round 1.

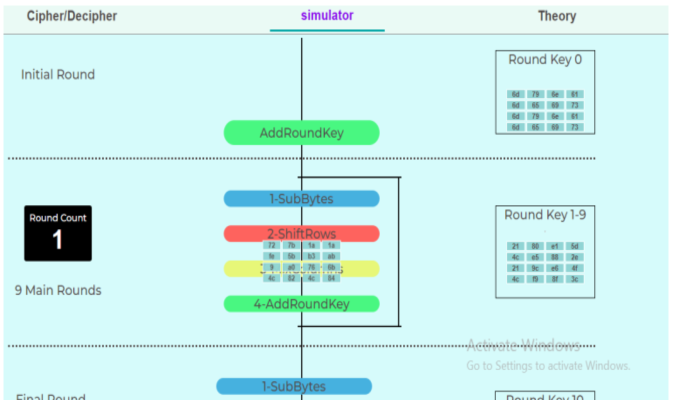


Fig 4.2.3: Events in a round

The initial round array meets various events, and undergoes transformations. SubBytes implements substitution using a lookup table also called the S-box, in next event, rows of the incoming 4*4 array are shifted certain times to left, MixColumns event does matrix multiplication and in final event Add Round Keys the resultant output of the previous stage is XOR-ed with the corresponding round key.

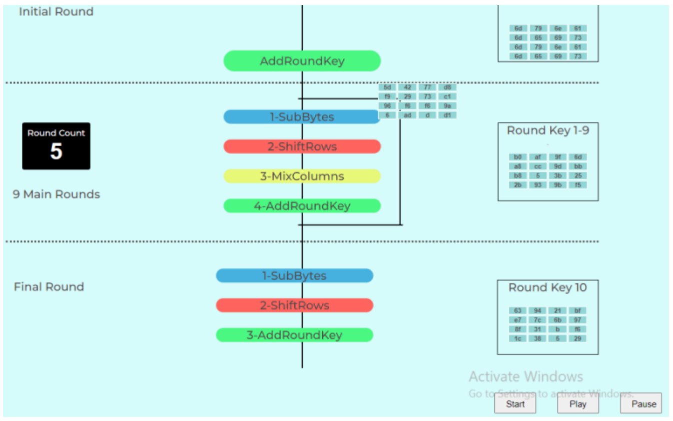


Fig 4.2.4: Repetition of rounds

All the above events are repeated again in a cyclic loop 9 times, the output photo shows how the array of add round key event went back as input to SubBytes. A counter is set to keep a track of the number of rounds ongoing.

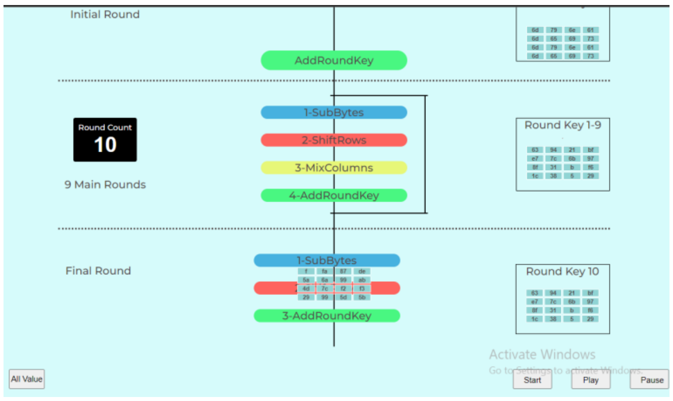


Fig 4.2.5: Final round of AES

After all the in between process changes, final round output is generated and the output shows how the array is flowing through the process, with no MixColumns event.

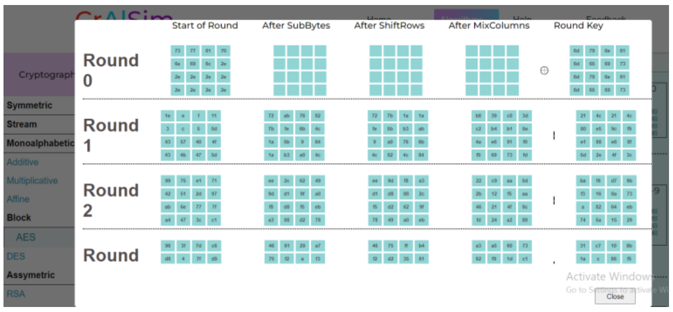


Fig 4.2.6: Calculation table

All intermediary calculations of events in each round are shown in the All-Value page. Each XOR result, along with mathematical operations of left shifts, matrix multiplication and substitutions can be seen in depth.

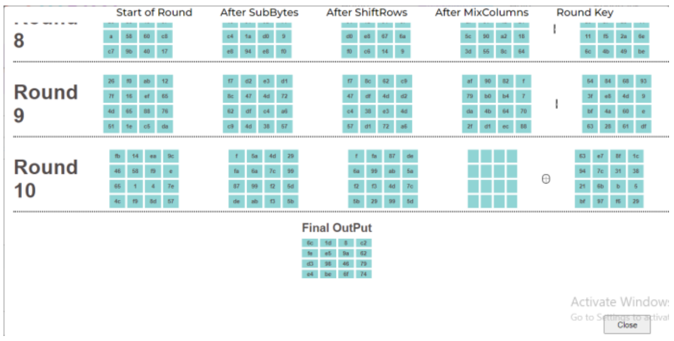
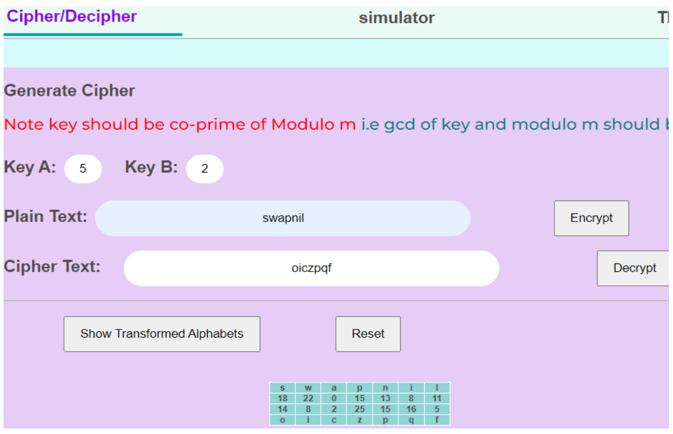


Fig 4.2.7: Final Output

Final output value is generated in a 4*4 array of hexadecimal value, which was converted back to base64 and shown in a proper string to user as the encrypted value in module one. As the output size remains same for an encrypted data, it can be compared to hashing which is irreversible, and helps in preventing possible issues of two encrypted strings having same input [16].

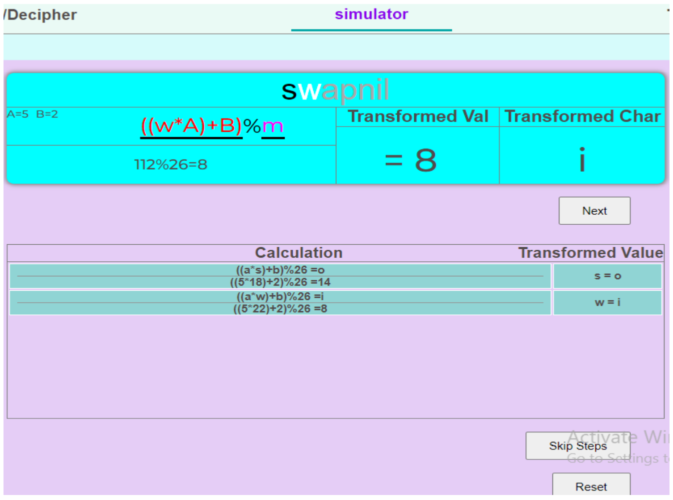
Affine:



s	w	a	p	n	i	l
18	22	0	15	13	8	11
14	8	2	25	15	16	5
o	i	c	z	p	q	f

Fig 4.2.8: Affine MAIN PAGE

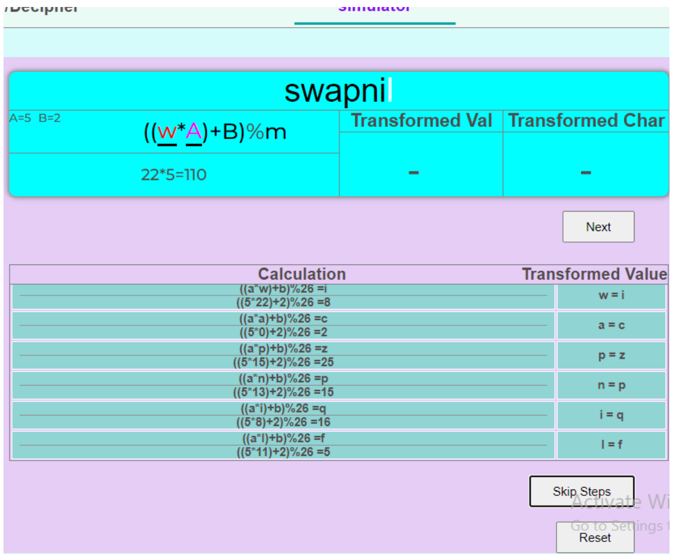
Key A is 5 and key B is given 2 as input, they satisfy the condition of being co-prime. Plain text is provided, and cipher is generated, and all transformed alphabets are represented in correspondence to their numeric values with letter ‘a’ starting from 0.



Calculation	Transformed Value
((a*s)+b)%26 =o ((5*18)+2)%26 =14	s = o
((a*w)+b)%26 =i ((5*22)+2)%26 =8	w = i

Fig 4.2.9: Simulator for Affine

For plain text ‘swapnil’, each letter [19] is fed into the formula (ax+b) %m, and subsequent calculations are shown each time Next button is clicked. Calculation for letter ‘s’ is done, so it’s highlighted in black, current letter under process is ‘w’ as it is seen in color white and yet to converted characters in gray. Every derivation and its resultant value are shown in transformation table with their calculations and changes implied.



Calculation	Transformed Value
((a*w)+b)%26 =i ((5*22)+2)%26 =8	w = i
((a*a)+b)%26 =c ((5*0)+2)%26 =2	a = c
((a*p)+b)%26 =z ((5*15)+2)%26 =25	p = z
((a*n)+b)%26 =p ((5*13)+2)%26 =15	n = p
((a*i)+b)%26 =q ((5*8)+2)%26 =16	i = q
((a*l)+b)%26 =f ((5*11)+2)%26 =5	l = f

Fig 4.2.10: Skip Steps functionality

If the process is comprehended for one letter, going through each letter transformation can be tedious, hence skip steps leads us to eventual values and the output is completely generated with affine cipher.

5. CONCLUSIONS

This paper focused on the proposed visualization tool for cryptographic algorithms, and elaborately exhibited the graphical view of the internal working of a few types. There are several encryption tools available over the internet, but not many visualization tools for these algorithms. No relational database is used to store the simulation steps. The factor which makes CrAlSim advantageous is that the algorithms covered are the primary ones in their categories, like Affine in monoalphabetic cipher and AES in block cipher, simple demonstration is also made for RSA [3], and that the complex calculations like the initial round with no event and 8 rounds of 4 distinguished events in a loop for 128 bits key in AES all can be visualized and understood rather than students mugging up theory of each step. The existing simulator on further upgrade can be imbibed with more enhancements like adding more in-depth simulations of the multiple rounds and events, adding a video based feature to download a mp4 file of an instance simulation, adding comparison factors to two similar inputs but different ciphers generated as output, and also more highlighting of certain events in color scheme as what to visualize is different from the perception of each user.

REFERENCES

[1] L. Ogiela, M. R. Ogiela and U. Ogiela, "Cognitive information systems in secure information management and personalized cryptography," The city of Kitakyushu hosted the 2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and the 15th International Symposium on Advanced Intelligent Systems (ISIS), Japan, 2014, pp. 1152-1157, doi: 10.1109/SCIS-ISIS.2014.7044798

[2] U. Arom-oon, "An AES cryptosystem for small scale network," 2017 Third Asian Conference on Defence Technology (ACDT), Phuket, Thailand, 2017, pp. 49-53, doi: 10.1109/ACDT.2017.7886156.

[3] P. M. Aiswarya, A. Raj, D. John, L. Martin and G. Sreenu, "Binary RSA encryption algorithm," 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kumaracoil, India, 2016, pp. 178-181, doi: 10.1109/ICCICCT.2016.7987940.

[4] N. Buckley, A. Nagar and S. Arumugam, "Evolution of Visual Cryptography Basis Matrices with Binary Chromosomes," 2013 8th EUROSIM Congress on Modelling and Simulation, Cardiff, UK, 2013, pp. 7-12, doi: 10.1109/EUROSIM.2013.12.

[5] C. R. Martin, G. G. Trabes and G. A. Wainer, "A New Simulation Algorithm for PDEVS Models with Time Advance Zero," 2020 Winter Simulation Conference (WSC), Orlando, FL, USA, 2020, pp. 2208-2220, doi: 10.1109/WSC48552.2020.9384028.

[6] A. Kabulov and M. Berdimurodov, "Optimal representation in the form of logical functions of microinstructions of cryptographic algorithms (RSA, El-Gamal)," 2021 International Conference on Information Science and Communications Technologies (ICISCT), Tashkent, Uzbekistan, 2021, pp. 1-4, doi: 10.1109/ICISCT52966.2021.9670149.

[7] R. I. Masya, R. F. Aji and S. Yazid, "Comparison of Vigenere Cipher and Affine Cipher in Three-pass Protocol for Securing Image," 2020 6th International Conference on Science and Technology (ICST), Yogyakarta, Indonesia, 2020, pp. 1-5, doi: 10.1109/ICST50505.2020.9732873.

[8] R. Mahendran and K. Mani, "Generation of Key Matrix for Hill Cipher Encryption Using Classical Cipher," 2017 World Congress on Computing and Communication Technologies (WCCCT), Tiruchirappalli, India, 2017, pp. 51-54, doi: 10.1109/WCCCT.2016.22.

[9] M. Kocheta, N. Sujatha, K. Sivakanya, R. Srikanth, S. Shetty and P. V. Ananda Mohan, "A review of some recent stream ciphers," 2013 International conference on Circuits, Controls and Communications (CCUBE), Bengaluru, India, 2013, pp. 1-6, doi: 10.1109/CCUBE.2013.6718558.

[10] Y. Li, H. Zhang, H. Ju and X. Jiang, "Visual Simulation Technique of Decision Making of Interactive Stand Management Methods," 2014 International Conference on Virtual Reality and Visualization, Shenyang, China, 2014, pp. 459-466, doi: 10.1109/ICVRV.2014.37.

[11] Bloom, Yuval & Fields, Ilai & Maslennikov, Alona & Rozenman, Georgi. (2022). Quantum Cryptography—A Simplified Undergraduate Experiment and Simulation. Physics. 4. 104-123. 10.3390/physics4010009.

[12] Walter Tuchman (1997). "A brief history of the data encryption standard". Internet besieged: countering cyberspace scofflaws. ACM Press/Addison-Wesley Publishing Co. New York, NY, USA. pp. 275–280.

[13] Gurpreet K. and Vishal A., (2013), “An Efficient Implementation of RSA Algorithm using FPGA and Big Prime Digit”, International Journal of Computer & Communication Engineering Research (IJCCER), Volume 1 - Issue 4.

[14] Alemami, Yahia & Mohamed, Mohamad A & Atiewi, Saleh. (2023). Advanced approach for encryption using advanced encryption standard with chaotic map. International Journal of Electrical and Computer Engineering (IJECE). 13. 10.11591/ijece.v13i2.pp1708-1723.

[15] A. Singh, “Comparatively analysis of AES, DES and SDES algorithms [9]” In 2014, the International Journal of Computer Science and Technology published a research paper titled "Comparative analysis of encryption algorithms for various types of data files for data security," authored by K. P. Karule and N. V. Nagrale. The paper was published in volume 8491, issue 3, and spanned pages 200-202.

[16] M. M. Bachtiar, T. H. Ditanaya, S. Wasista, and R. R. K. Perdana, “Security enhancement of AES based encryption using dynamic salt algorithm,” in Proceedings of the 2018 International Conference on Applied Engineering, ICAE 2018, 2018, pp. 1–6, doi: 10.1109/INCAE.2018.8579381.

[17] M. I. Reddy, “A modified advanced encryption standard algorithm,” Journal of Mechanics of Continua and Mathematical Sciences, no. 1, pp. 112–117, 2020, doi: 10.26782/jmcms.spl.5/2020.01.00027.

[18] A. Hafsa, A. Sghaier, M. Zeghid, J. Malek, and M. Machhout, “An improved co-designed AES-ECC cryptosystem for secure data transmission,” in International Journal of Information and Computer Security, 2020, vol. 13, no. 1, pp. 118–140, doi: 10.1504/IJICS.2020.108145.

[19] Arroyo, Jan Carlo & Delima, Allemar Jhone. (2020). A Keystream-Based Affine Cipher for Dynamic Encryption. International Journal of Emerging Trends in Engineering Research. 8. 2919-2922. 10.30534/ijeter/2020/06872020.