Chaotic Image Encryption Techniques

Prof. Dr. Sudan Jha
School of Computer Engineering, Campus 15
KIIT University, Patia Bhubaneswar - 751024

Abstract- The development of the Space Science and Technology has recently attracted a growing interest from researchers and industrial communities, mainly because of large number of possible applications capable to exploit remotely sensed data and satellite images. Advances in space science, data analysis, and communication technologies present new opportunities for users to increase productivity, reduce costs, facilitate innovation and create virtual collaborative environments for addressing the new challenges. GIS and Remote sensing technologies, along with related geospatial technologies, contribute powerful tools for preserving and protecting the nation‟s critical infrastructure.

In such systems, a space borne platform collects scientific data and transmits them to a ground station and at the ground segment, a series of image products are created that can be made available to research or commercial organizations for exploitation. The data delivery and sharing process, usually based on CD/DVD-ROM or on shared network environment (Internet, LAN, WAN etc), provides the user with a digital version of the remote sensing data and images. In the same way as for multimedia contents, the digital format implies an inherent risk of unauthorized copy or use of the product.

Similarly many digital services, such as Medical, Military, and Space imaging systems require reliable security in storage and transmission of digital images. The rapid progress of Internet in the digital world today, the security of digital images has become more and more important. The prevalence of multimedia technology in our society has promoted digital images to play a more significant role, which demands a serious protection of users‟ privacy. To fulfill such security and privacy needs in various applications, encryption of images is very important to minimize malicious attacks from unauthorized parties.

Keywords – Network, Satellite Images, Internet, Networking, Digital Images, Encryption

I. INTRODUCTION

This research work is intended to the people who need privacy for their confidential images. It was most suitable in the networking Systems, so it was more eligible in Space Science research centers. Although there are many image encryption techniques none of them are suitable for the networking systems. So the main scope of this work is to provide security for the images in the networking systems. My work provides safe ways of means to transfer images between the networking systems confidentially.

There are many scientists who worked on the chaotic encryption techniques like ***S. Bennie et al***. proposed a novel algorithm for image encryption based on mixture of chaotic maps. In this symmetric key cryptography technique, a typical coupled map was mixed with a one-dimensional chaotic map and used for high degree security image encryption while its speed is acceptable. This mixture application of chaotic maps shows advantages of large key space and high-level security. The cipher generated by this method is the same size as the plaintext. Similarly in, ***Jiri Fridrich*** showed that how to adapt certain invertible chaotic two-dimensional maps on a torus or on a square to create symmetric block encryption schemes.

The schemes are especially useful for encryption of large amount of data, such as digital images or electronic databases. In this a chaotic map is first generalized by introducing parameters and the discretized to finite square lattice of points which represent pixels or some other data items. The discretized map is further extended to three dimensions and composed with simple diffusion mechanism. As a result, a block product encryption scheme is obtained. To encrypt an N x N image, the ciphering map is iteratively applied to the image. The main features of the encryption scheme studied in are: a variable key length, a relatively large block size (several KB or more), and a high encryption rate (1Mb un-optimized C code on a 60MHz Pentium). The cipher is based on two dimensional chaotic maps, which are used for creating complex, key-dependent permutations. Unlike most today's symmetric encryption schemes, which rely on complex substitution rules while somewhat neglecting the role of permutations, the cipher is based on complex permutations composed with a relatively simple diffusion mechanism. A somewhat different chaotic maps based image encryption method was proposed in to improve the properties of confusion and diffusion in terms of discrete exponential chaotic maps, and design a key scheme for the resistance to statistic attack, differential attack and grey code attack be-cause of floating-point analytical computation outcome of chaotic system,

unauthorized users or attacker can attack cryptosystems and effectively via certain basin structures. To increase security level of cryptosystem, they combined approaches of chaotic and conventional crypto-graphic methods in some ways as in. To achieve diffusion and resistance from differential attack, they analyze discrete exponential chaotic map performance and de-sign the permutation of the pixels of image after that performs XOR plus mod-operation on designed permutation.

Similarly, to achieve and increasing resistance level from statistic attack, grey code attack and entropy attack, Linhua et al. chose chaotic map elaborately and construct chaotic sequences which satisfy uniform distribution. To achieve high encryption speed, in the early stages some elementary cryptographic methods using algebra-based transformation scrambling techniques were suggested. Since the operations are simple, the encryption does not require high computational cost but recent research suggest that Chaos-based transformations naturally have affine linearity while algebra-based transformations have fixed scrambling matrixes for data encryption. The cryptanalysis shows that the fixed or linear scrambling matrices are insecure for image encryption.

Similar thoughts also appeared in by Wang et al., where a Poker shuffle is used for scrambling image, which is controlled dynamically by chaotic system. This scrambling technique belongs to the position permutation and it has several features like nonlinearity, non-analytic formula and large key space. In addition, it can deal with non-square image while many scrambling methods with analytic formula only process square image and can be integrated into the existing image encryptions and digital image water-marking. To overcome these limitations in Logistic maps the nonlinear chaotic algorithm (NCA) map uses power function and tangent function instead of linear function. Many chaotic circuits and their application schemes have been proposed to secure communications since Pecora and Carrol proposed a method to synchronize two identical chaotic systems. The chaotic secure system can be used in applications that do not require a high level of information security such as remote keyless entry system, video phone, and wireless telephone. Some existing Hard-ware based chaotic secure communication schemes have been briefly reviewed in this Section.

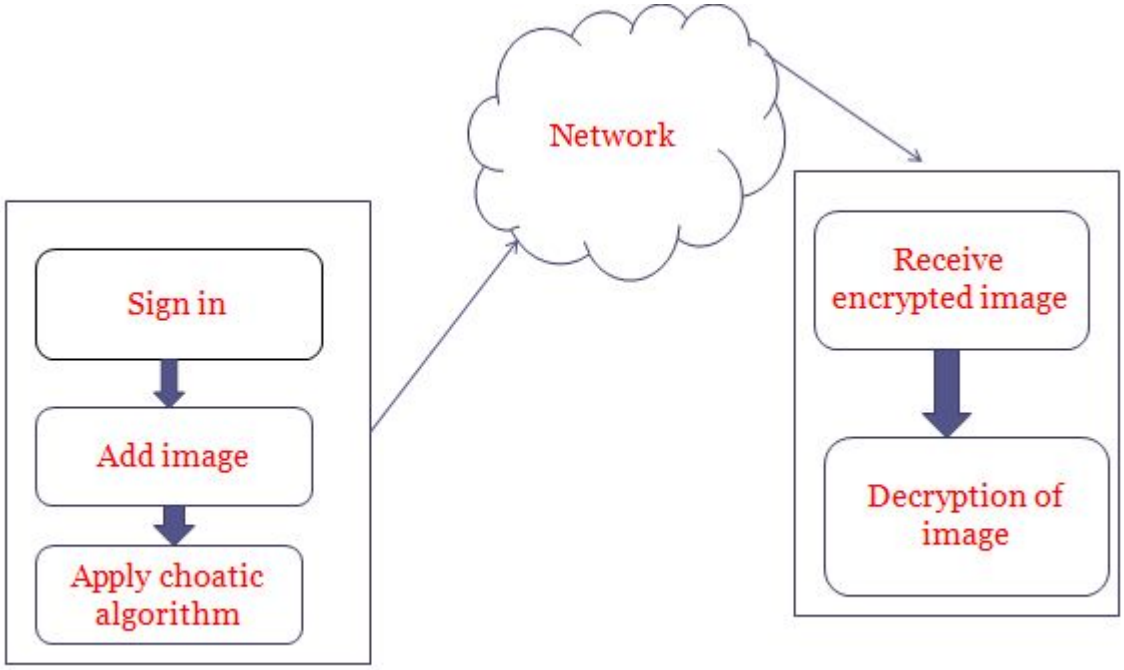

Figure 1 Basic Architecture

II. PROPOSED METHOD

2.1 Proposed System –

Proposed system provides a solution to existing system by extending its facilities. The proposed study aims to explore the possibility of using chaotic or chaos-based encryption techniques to protect remote sensing satellite images and provides high level of security in efficient and reliable way.

Chaos based cryptographic scheme provides high security level, less computational time and power in reliable and efficient way to deal with balky, difficult and intractable data that why many researchers recommends that it is more

suitable for multimedia data, especially for images. Chaos-based system have many properties to achieve high security level, such as sensitivity to change initial conditions and parameters, periodicity (a system that tends in probability to a limiting form that is independent of the initial conditions), random behavior and unstable periodic orbits with long periods. It has very high diffusion and confusion properties that are desirable for cryptosystem.

1. Maintenance is easy.
2. Provides high security level.
3. This project takes less computational time and power in reliable and efficient way to deal with balky, difficult & intractable data.
4. It is more suitable for multimedia data, especially for images.
5. This project has many properties to achieve high security level, such as sensitivity to change initial conditions and parameters, periodicity, random behavior and unstable periodic orbits with long periods.
6. It has very high diffusion and confusion properties that are desirable for cryptosystem.
7. Encrypting and Decrypting of the image is very easy.
8. Sending or transferring the image via the network is easier, just to start server class at receiver side.
9. By entering the key with which the key was encrypted we can get the original image in the other system.
10. Viewing the status of encryption and decryption processes will also be there.

2.2 System Features and Functionalities –

The user can have membership by concerning the administrator. The system dynamically generates member id on presenting the user details. The user can have transactions by having their member id. The user adds the image for encryption and encrypts it for confidential images.

2.3 Process Modules –

Software engineering is an engineering discipline whose goal is the cost-effective development of software systems. Software engineering is concerned with development and maintenance of technological products, problem solving techniques common to all engineering disciplines. A development strategy that encompasses the process, methods and tools layers is referred to software engineering paradigm or process model. A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used and the controls and derivable those are required.

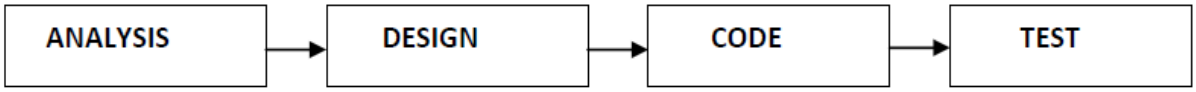

Figure 2 Linear Sequential Model

As software is always a part of a larger system, work establishing requirements for all system elements and then allocating some subset of these requirements to software. System engineering and analysis encompass requirements gathering at the system level with a small amount of top-level design and analysis. Information engineering encompasses requirements gathering at the strategic business level and at the business area level. The requirements gathering process is intensified and focused specifically on software. Software design is actually a multi-step process that focuses on four distinct attributes of a program: data structure, software architecture, interfaces representation a procedural detail. The design process translates requirements into a representation of the software that can be accessed for quality before coding begins. Like requirements, the design is documented and becomes a part of the software configuration. The design is been translated into a machine-readable form. The code generation step performs this task. If the design is performed in a detailed manner, code generation can be performed mechanically. The process focuses on the logical internals of software, ensuring that all statements have been tested and on the functional externals, that is conducting tests to uncover errors and ensure that defined input will produce actual results that agree with required results. As the linear sequential model is the oldest and the most widely used paradigm for software engineering, and can be used efficiently for small projects that require less customer communication, it has been in development of data security project.

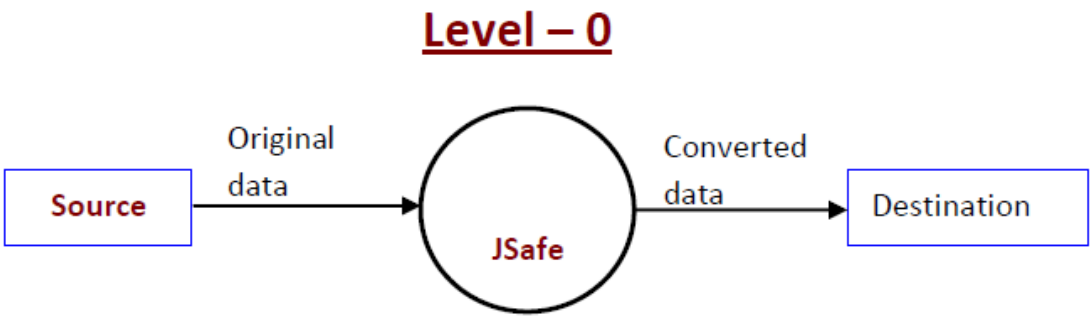

Figure 3 Context Level 0 of the Data Flow Diagram

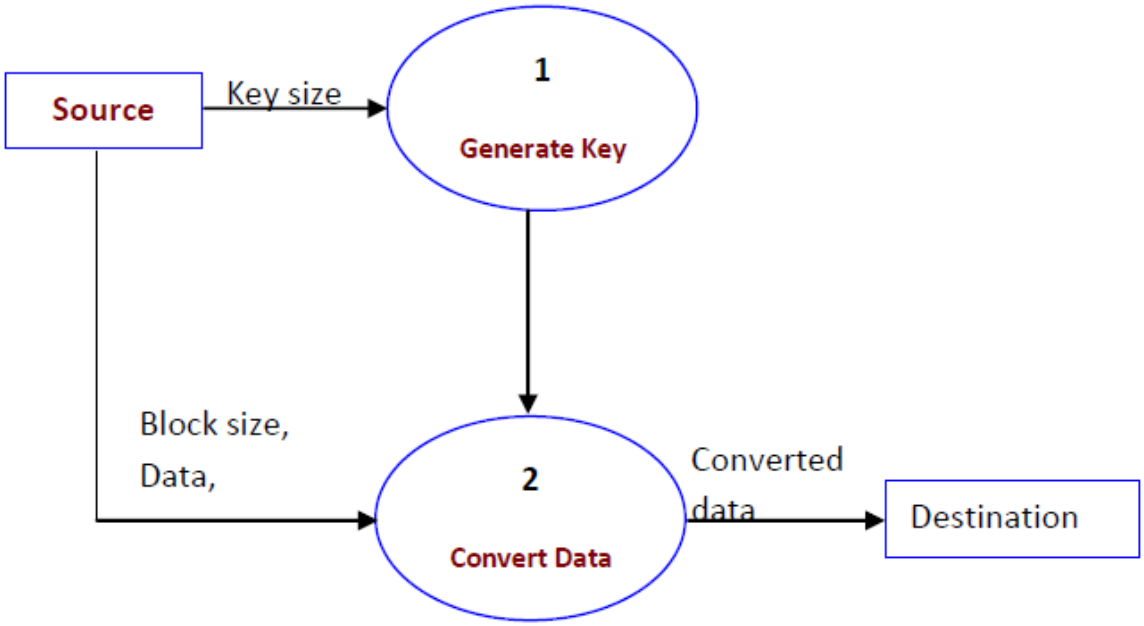

Figure 4 Context Level 1 of the Data Flow Diagram

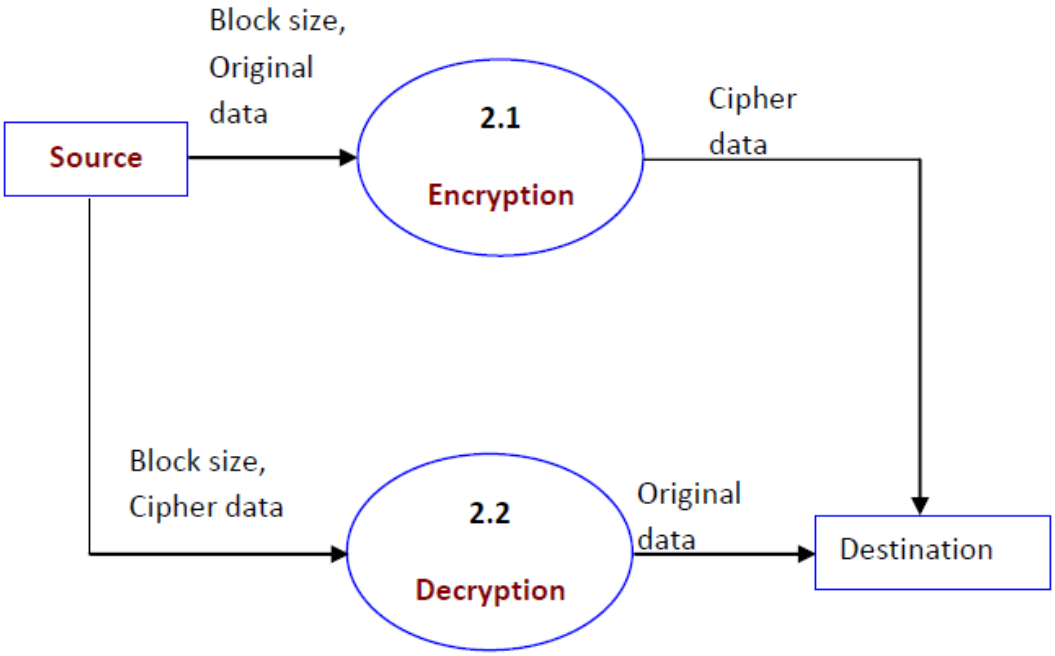

Figure 5: Context Level 2 of the Data Flow Diagram

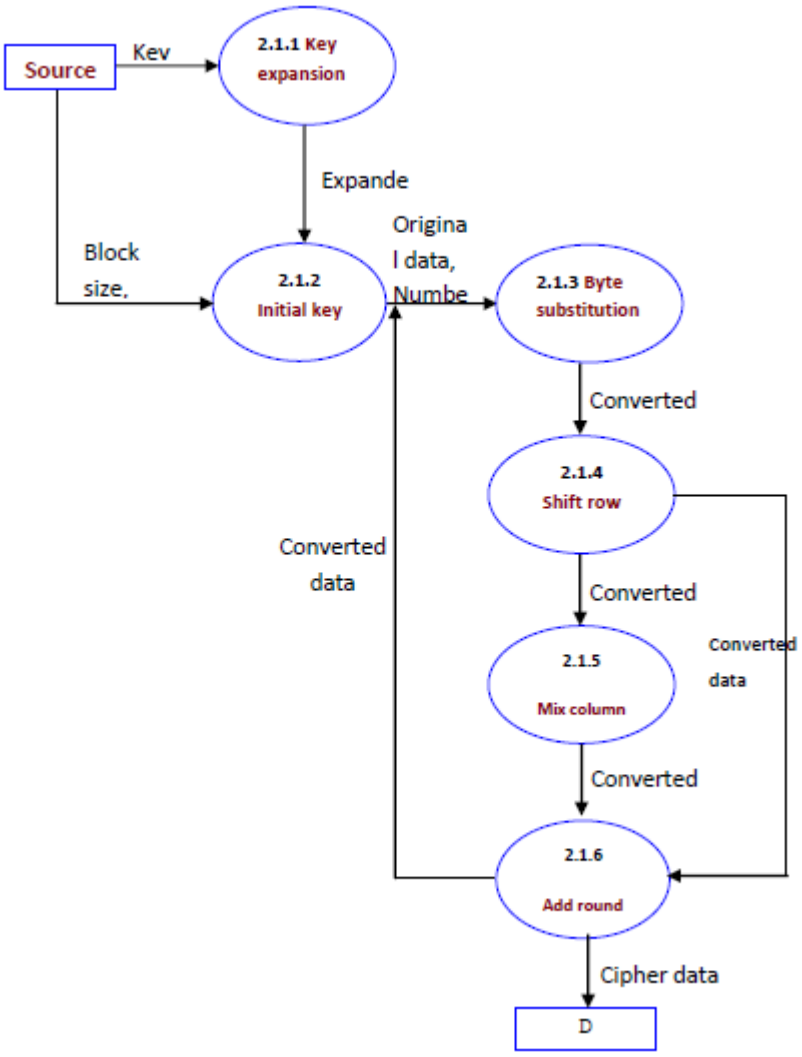

Figure 6: Context Level 3 of the Data Flow Diagram

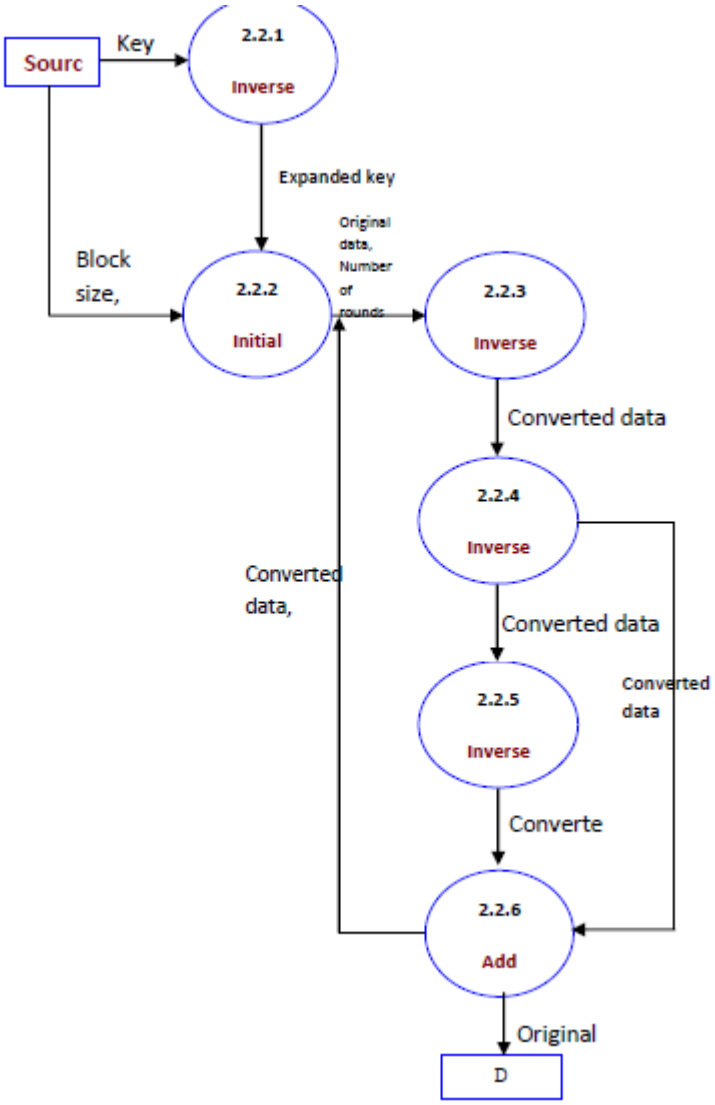

Figure 7: Context Level 4 of the Data Flow Diagram

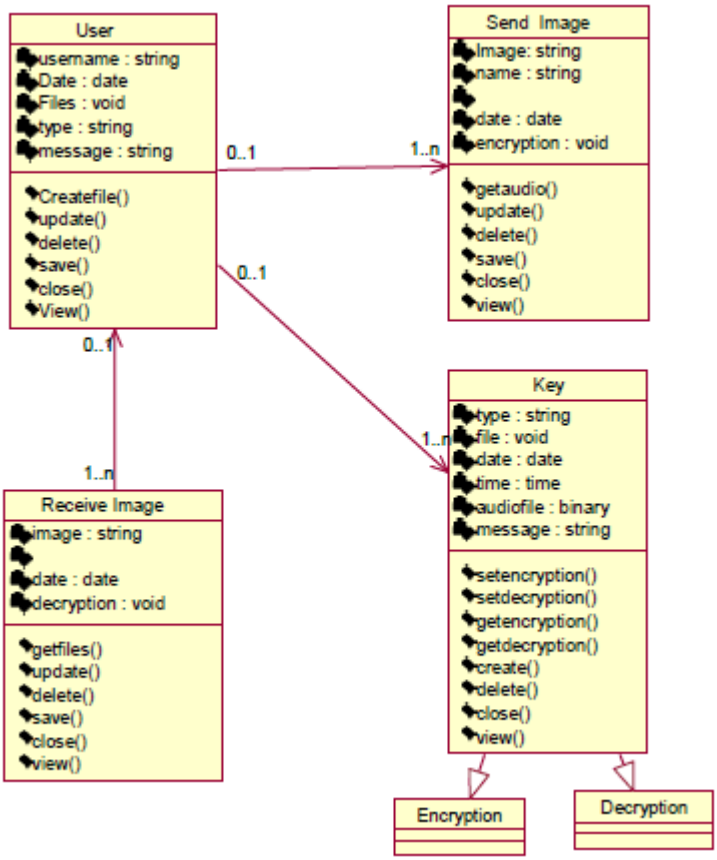

Figure 8: Class Diagram

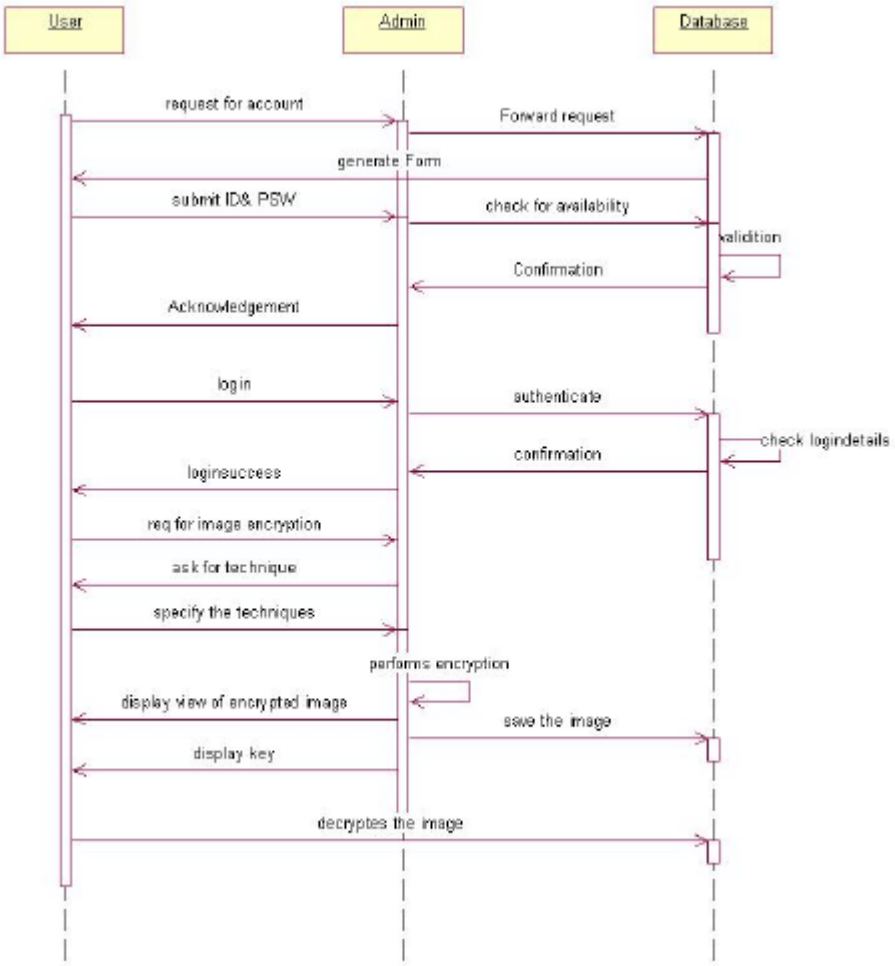

Figure 9: Sequence Diagram

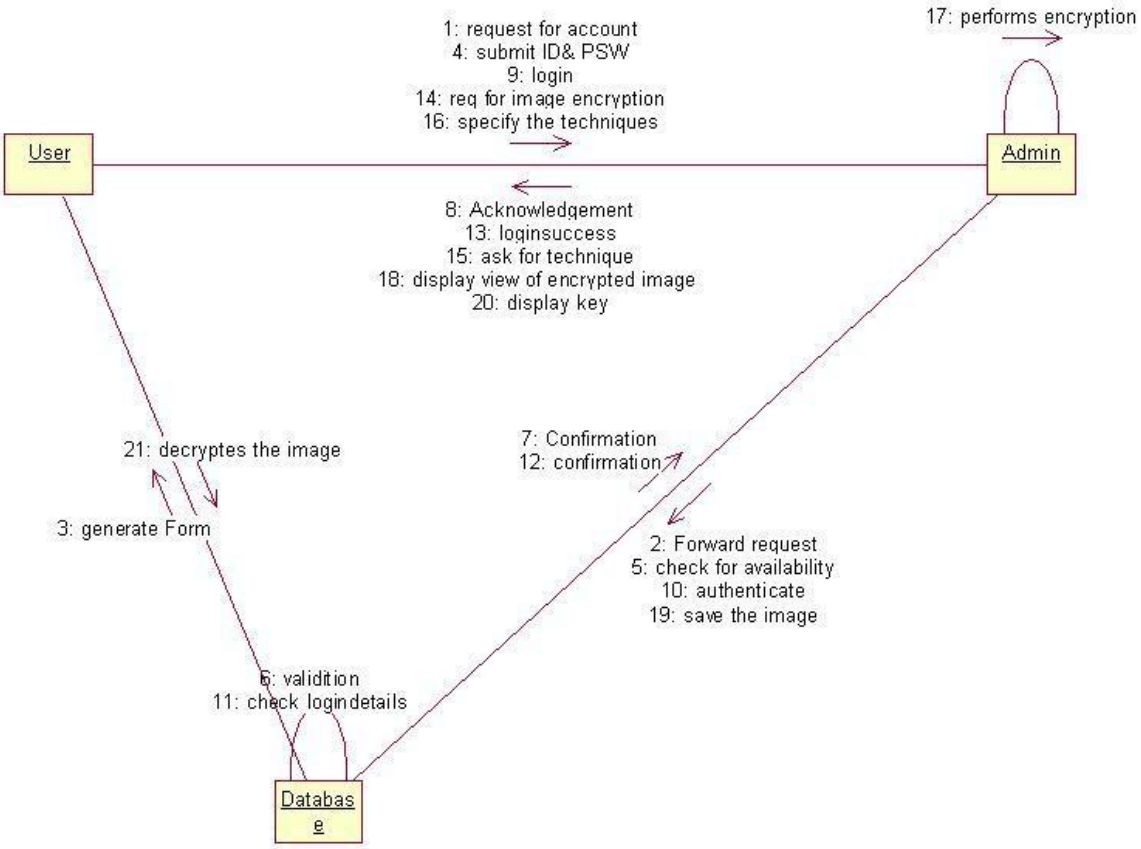

Figure 10: The overall Collaboration Diagram

III. PROPOSED ALGORITHM AND IMPLEMENTATION

3.1 Blowfish–

Blowfish is a variable-length key block cipher. It does not meet all the requirements for a new cryptographic standard discussed above: it is only suitable for applications where the key does not change often, like a communications link or an automatic file encryptor. It is significantly faster than DES when implemented on 32-bit microprocessors with large data caches, such as the Pentium and the PowerPC. Blowfish is a variable-length key, 64-bit block cipher. The algorithm consists of two parts: a key-expansion part and a data- encryption part. Key expansion converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes.

3.2 Description of Algorithm –

Data encryption occurs via a 16-round Feistel network. Each round consists of a key-dependent permutation, and a key- and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookups per round.

3.2.1 Subkeys –

Blowfish uses a large number of subkeys. These keys must be precomputed before any data encryption or decryption.

1. The P-array consists of 18 32-bit subkeys: P1, P2,..., P18.

2. There are four 32-bit S-boxes with 256 entries each: S1,0, S1,1,..., S1,255; S2,0, S2,1,..,, S2,255; S3,0, S3,1,..., S3,255; S4,0, S4,1,..,, S4,255. The exact method used to calculate these subkeys will be described later.

3.2.2 Encryption –

1. Blowfish is a Feistel network consisting of 16 rounds (see Figure 1). The input is a 64-bit data element, x.
2. Divide x into two 32-bit halves: xL, xR For i = 1 to 16: xL = xL XOR Pi xR = F(xL) XOR xR Swap xL and xR Swap xL and xR (Undo the last swap.) xR = xR XOR P17 xL = xL XOR P18 Recombine xL and xR
3. Function F Divide xL into four eight-bit quarters: a, b, c, and d F(xL) = ((S1,a + S2,b mod 232) XOR S3,c) + S4,d mod 232

Decryption is exactly the same as encryption, except that P1, P2,..., P18 are used in the reverse order. Implementations of Blowfish that require the fastest speeds should unroll the loop and ensure that all subkeys are stored in cache.

3.2.3 Generating the Subkeys –

The subkeys are calculated using the Blowfish algorithm. The exact method is as follows:

1. Initialize first the P-array and then the four S-boxes, in order, with a fixed string. This string consists of the hexadecimal digits of pi (less the initial 3). For example:

P1 = 0x243f6a88 P2 = 0x85a308d3 P3 = 0x13198a2e P4 = 0x03707344

2. XOR P1 with the first 32 bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (possibly up to P14). Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits. (For every short key, there is at least one equivalent longer key; for example, if A is a 64-bit key, then AA, AAA, etc., are equivalent keys.)

3. Encrypt the all-zero string with the Blowfish algorithm, using the subkeys described in steps (1) and (2).

4. Replace P1 and P2 with the output of step (3).

5. Encrypt the output of step (3) using the Blowfish algorithm with the modified subkeys.

6. Replace P3 and P4 with the output of step (5).

7. Continue the process, replacing all entries of the P- array, and then all four S-boxes in order, with the output of the continuously-changing Blowfish algorithm.

In total, 521 iterations are required to generate all required subkeys. Applications can store the subkeys rather than execute this derivation process multiple times.

3.3 MINI-BLOWFISH –

The following mini versions of Blowfish are defined solely for cryptanalysis. They are not suggested for actual implementation. Blowfish-32 has a 32-bit block size and subkey arrays of 16-bit entries (each S-box has 16 entries). Blowfish-16 has a 16-bit block size and subkey arrays of 8-bit entries (each S-box has 4 entries).

3.4 DESIGN DECISIONS –

The underlying philosophy behind Blowfish is that simplicity of design yields an algorithm that is both easier to understand and easier to implement. Through the use of a streamlined Feistel network--a simple S-box substitution and a simple P-box substitution--I hope that the design will not contain any flaws. A 64-bit block size yields a 32-bit word size, and maintains block-size compatibility with existing algorithms. Blowfish is easy to scale up to a 128-bit block, and down to smaller block sizes. Cryptanalysis of the mini-Blowfish variants may be significantly easier than cryptanalysis of the full version. The fundamental operations were chosen with speed in mind. XOR, ADD, and MOV from a cache are efficient on both Intel and Motorola architectures. All subkeys fit in the cache of a 80486, 68040, Pentium, and PowerPC. The Feistel network that makes up the body of Blowfish is designed to be as simple as possible, while still retaining the desirable cryptographic properties of the structure. Figure 3 is round i of a general Feistel network: Rn,i are reversible functions of text and key, and Ni is a non-reversible function of text and key. For speed and simplicity, I chose XOR as my reversible function. This let me collapse the four XORs into a single XOR, since:

R--1,i+1 = R1,i+1 XOR R2,i-1 XOR R3,i XOR R4,i

This is the P-array substitution in Blowfish. The XOR can also be considered to be part of the non-reversible function, Ni, occurring at the end of the function. (Although equivalent, I chose not to illustrate them in this way because it simplifies description of the subkey-generation process.) There are two XORs that remain after this reduction: R1 in the first round and R2 in the last round. I chose not to eliminate these in order to hide the input to the first non-reversible function. We considered a more complicated reversible function, one with modular multiplications and rotations. However, these operations would greatly increase the algorithm's execution time. Since function F is the primary source of the algorithm's security, I decided to save time-consuming complications for that function.

Function F, the non-reversible function, gives Blowfish the best possible avalanche effect for a Feistel network: every text bit on the left half of the round affects every text bit on the right half. Additionally, since every subkey bit is affected by every key bit, the function also has a perfect avalanche effect between the key and the right half of the text after every round. Hence, the algorithm exhibits a perfect avalanche effect after three rounds and again every two rounds after that. We considered adding a reversible mixing function, more complicated than XOR, before the first and after the last round. This would further confuse the entry values into the Feistel network and ensure a complete avalanche effect after the first two rounds. I eventually discarded the addition as a time- consuming complication with no clear cryptographic benefits. The non-reversible function is designed for strength, speed, and simplicity. The small-number of bits to large-number of bits may have weaknesses with respect to linear cryptanalysis, but these weaknesses are hidden both by combining the output of four S-boxes and making them dependent on the key.

We used four different S-boxes instead of one S-box primarily to avoid symmetries when different bytes of the input are equal, or when the 32-bit input to function F is a bytewise permutation of another 32-bit input. I could have used one S-box and made each of the four different outputs a non-trivial permutation of the single output, but the four S-box design is faster, easier to program, and seems more secure. The function that combines the four S-box outputs is as fast as possible. A simpler function would be to XOR the four values, but mixing addition mod 232 and XOR combines two different algebraic groups with no additional instructions. The alternation of addition and XOR ends with an addition operation because an XOR combines the final result with xR.

If the four indexes chose values out of the same S-box, a more complex combining function would be required to eliminate symmetries. I considered using a more complex combining function in Blowfish (using modular multiplications, rotations, etc.), but chose not to because the added complication seemed unnecessary. The key-dependent S-boxes protect against differential and linear cryptanalysis. Since the structure of the S-boxes is completely hidden from the cryptanalyst, these attacks have a more difficult time exploiting that structure. While it would be possible to replace these variable S-boxes with four fixed S-boxes that were designed to be resistant to these attacks, key-dependent S-boxes are easier to implement and less susceptible to arguments of "hidden" properties. Additionally, these S-boxes can be created on demand, reducing the need for large data structures stored with the algorithm.

Each bit of xL is only used as the input to one S-box. In DES many bits are used as inputs to two S-boxes, which strengthen the algorithm considerably against differential attacks. I feel that this added complication is not as necessary with key- dependent S-boxes. Additionally, larger S-boxes would take up considerably more memory space. Function F does not depend on the iteration. I considered adding this dependency, but did not feel that it had any cryptographic merit. The P-array substitution can be considered to be part of this function, and that is already iteration-dependent. The number of rounds is set at 16 primarily out of desire to be conservative. However, this number affects the size of the P- array and therefore the subkey-generation process; 16 iterations permits key lengths up to 448 bits. I expect to be able to reduce this number, and greatly speed up the algorithm in the process, as I accumulate more cryptanalysis data.

In algorithm design, there are two basic ways to ensure that the key is long enough to ensure a particular security level. One is to carefully design the algorithm so that the entire entropy of the key is preserved, so there is no better way to crypt analyzes the algorithm other than brute force. The other is to design the algorithm with so many key bits that attacks that reduce the effective key length by several bits are irrelevant. Since Blowfish is designed for large microprocessors with large amounts of memory, I chose the latter.

The subkey generation process is designed to preserve the entire entropy of the key and to distribute that entropy uniformly throughout the subkeys. It is also designed to distribute the set of allowed subkeys randomly throughout the domain of possible subkeys. I chose the digits of pi as the initial subkey table for two reasons: because it is a random sequence not related to the algorithm, and because it could either be stored as part of the algorithm or derived when needed. There is nothing sacred about pi; any string of random bits--digits of e, RAND tables, and output of a random number generator--will suffice. However, if the initial string is non-random in any way (for example, ASCII text with

the high bit of every byte a 0), this non-randomness will propagate throughout the algorithm. In the subkey generation process, the subkeys change slightly with every pair of subkeys generated. This is primarily to protect against any attacked of the subkey generation process that exploit the fixed and known subkeys. It also reduces storage requirements. The 448 limit on the key size ensures that the every bit of every subkey depends on every bit of the key. (Note that every bit of P15, P16, P17, and P18 does not affect every bit of the cipher text, and that any S-box entry only has a .06 probability of affecting any single cipher text block.)

The key bits are repeatedly XORed with the digits of pi in the initial P-array to prevent the following potential attack: Assume that the key bits are not repeated, but instead padded with zeros to extend it to the length of the P-array. An attacker might find two keys that differ only in the 64-bit value XORed with P1 and P2 that, using the initial known subkeys, produce the same encrypted value. If so, he can find two keys that produce all the same subkeys. This is a highly tempting attack for a malicious key generator.

To prevent this same type of attack, I fixed the initial plaintext value in the subkey-generation process. There is nothing special about the all-zeros string, but it is important that this value be fixed. The subkey-generation algorithm does not assume that the key bits are random. Even highly correlated key bits, such as an alphanumeric ASCII string with the bit of every byte set to 0, will produce random subkeys. However, to produce subkeys with the same entropy, a longer alphanumeric key is required. The time-consuming subkey-generation process adds considerable complexity for a brute-force attack. The subkeys are too long to be stored on a massive tape, so they would have to be generated by a brute-force cracking machine as required. A total of 522 iterations of the encryption algorithm are required to test a single key, effectively adding 29 steps to any brute-force attack.

IV. CONCLUSION

Advances in space science, data analysis, and communication technologies present new opportunities for users to increase productivity, reduce cost, facilitate innovation and create virtual collaborative environments for addressing new challenges. In such processes data sharing is usually based on CD/DVD-ROM hardcopy or on shared network environment (Internet, LAN, WAN etc) , so there exists inherent security risk of unauthorized access or use of the product. To fulfil such security and privacy needs in various applications, encryption of such data is very important to minimize malicious attacks from unauthorized parties and to safeguard sensitive data.

From the study of the above traditional and chaos-based image encryption techniques; it is quite clear that traditional image encryption techniques DES, Triple-DES and IDEA have some limitations such as these algorithms have high security level under CBC mode but requires large data size, long computational time and high computing power.

On the other hand chaos-based cryptographic scheme provides high security level, less computational time and power in reliable and efficient way to deal with balky, difficult and intractable data that why many researchers recommends that it is more suitable for multimedia data, especially for images . Chaos-based system have many properties to achieve high security level, such as sensitivity to change initial conditions and parameters, ergodicity (a system that tends in probability to a limiting form that is independent of the initial conditions), random behaviour and unstable periodic orbits with long periods. It has very high diffusion and confusion properties that are desirable for cryptosystem.

V. FUTURE SCOPE

This project represents a preliminary study of different mechanisms used for image protection highlighting comprehensive comparative overview of existing traditional image encryption techniques like DES, Triple-DES and IDEA algorithms. Some existing chaos-based image encryption schemes have also been briefly reviewed by studying and analyzing multiple algorithms properties, such as, encryption speed, compatibility to image format and compression standards, and real-time implementation etc. Now a days chaos-based encryption technique is getting more and more popularity worldwide to deal with multimedia data sets especially for images and is recommended by many researchers [7 -12]. The similar technique of chaos-based image encryption is planned to be applied on satellite imageries since chaos-based techniques have many proper-ties to achieve high security level in efficient and reliable manner, like sensitivity, ergodicity, randomness, and no recurring or reappearing orbits for long periods with high diffusion and confusion. The summary of the planed future work is the following:

Continue to study other proposed encryption techniques, theories and algorithms, such as chaos-based or chaotic cryptosystem, number theory, AES, DES, RSA etc., need to be understood.

Investigate and compare the security and robustness of some chaos-based and traditional encryption schemes for large data sets e.g. huge satellite images.

In-depth study general insecurity properties of satellite imagery and remotely sensed data, and try to propose some corresponding countermeasures, furthermore, de-sign some new schemes that have good security properties and can meet the real application requirements at the same time.

REFERENCES

[1]. Mauro Barni, Franco Bartolini, Enrico Magli and Gab-riella Olmo, "Watermarking techniques for electronic delivery of remote sensing images", Optical Engineer-ing, 41,No. 9, pp. 2111-2119, September 2002.

[2]. Ray A. Williamson, "REMOTE SENSING AND TRANSPORTATION SECURITY", Pecora 15/Land Satellite Information IV/ISPRS Commission I/FIEOS Conference Proceedings, 2002

[3]. Klaus Holtz, "Advanced Data Compression promises the next big Leap in Network Performance", IS&T / SPIE EUROPTO Conference, Zurich, Switzerland, May 1998.

[4]. R. J. Anderson and F. A. P. Petitcolas, "On the limits of steganography," IEEE Journal on Selected Areas in Communications, vol. 16, no. 4, pp. 474-481, 1998

[5]. Mohammad Zakir Hossain Sarker and Md. Shafiul Parvez, "A Cost Effective Symmetric Key Crypto-graphic Algorithm for Small Amount of Data", Pro-ceedings of the 9th IEEE International Multi topic Conference, pp. 1-6, December 2005

[6]. Xun Yi Chik How Tan Chee Kheong Slew Rahman Syed, M., "Fast encryption for multimedia," IEEE Transactions on Consumer Electronics, vol. 47, no. 1, pp. 101-107, 2001.

[7]. Yen J. C. and Guo J. I., "A new chaotic image encryp-tion algorithm," Proceeding of National Symposium on Telecommunications, pp. 358-362, December 1998.

[8]. Jui-Cheng Yen and J. I. Guo, "A New Chaotic Mirror-Like Image Encryption Algorithm and its VLSI Archi-tecture", Pattern Recognition and Image Analysis, vol.10, no.2, pp.236-247, 2000.

[9]. Jui-Cheng Yen and J. I. Guo, "Efficient Hierarchical Chaotic Image Encryption Algorithm and Its VLSI Re-alization". IEEE Proceeding Vis. Image Signal Process, vol. 147, no. 2, pp. 167-175,2000