Power Spectrum Encryption and Decryption of an Audio File

Sheetal Sharma, Himanshu Sharma and Lucknesh Kumar

Abstract: Encryption is a process of converting some data from its original form to encrypted form, practically that cannot understandable by unauthorized user. In this paper the power spectrum of an audio wav file is partially encrypted using RSA encryption technique. Here a transform domain audio signal (frequency domain audio signal) is taken for encryption and decryption. A time domain audio signal is converted to transform domain signal by using Fast Fourier Transform. A transform domain can separate a signal into different frequency regions with respective magnitude and phase values. All the frequency regions do not participate in communication equally, so we do not need to process whole frequency bins. We can apply our technique only on those regions which are useful for us. After analysis, we have selected the low frequencies which have higher magnitude values. Here, we can apply our scheme on higher magnitude values.

.

Keywords: Power Spectrum, Wav Signal, Transform Domain, Fast Fourier Transform.

I. INTRODUCTION

Data encryption is used to securely transmit information over an insecure channel. A mathematical transformation is applied to the information in such a way that it is very difficult to undo the transformation. A key, which is just a large number, controls the encryption [11] and decryption algorithm. A symmetric key algorithm uses the same key to encrypt and decrypt the data. An asymmetric key algorithm uses two different but related keys where one is used to encrypt and the other is used to decrypt the data [2]. Algorithms for data encryption tend to be computationally expensive especially on general-purpose processors that operate on words rather than single bits. The current standard DES algorithm operates in this fashion. Generally, the larger the size of the key the more computationally expensive the algorithm will be. A secure encryption algorithm assumes that an attacker knows everything about the system except for the key that is needed to decrypt the stolen information. Under ideal conditions, the only successful attack would be an exhaustive key search, in which the attacker must apply every possible key to the entire set of encrypted data, and then analyze the decrypted result to see if it was sensible. If this is the only possible attack, the system can be made secure in practice by choosing a large enough key size (i.e., the time required to test every possible key would be impractical).

Sheetal Sharma is a Research Scholar, GCET, Greater Noida, Himanshu Sharma is working as Assistant Professor, M. U., Aligarh, and Lucknesh Kumar is workind as Assistant Professor, GCET, Greater Noida, Emails: Sheetal.sys83@gmail.com, Himanshu.techfreak@gmail.com, lucknesh.mnnit@gmail.com

In many systems, a significant weakness can be used to reduce the time required to perform an exhaustive key search. This weakness occurs when the decrypted data is known to contain recognizable information. For example, if the encrypted data was known to be English text, statistics from the decrypted data could determine when something close to the English language was found [3]. This would indicate a potential key. If certain words were known to exist within the decrypted text, it would be even easier to find potential keys. In addition, it may be possible to perform a more efficient exhaustive key search if the encrypted data contains internal dependencies. If the data to be encrypted is a character, and the specific values of certain character elements determine the existence or size of subsequent character elements, these dependencies can be exploited by an attacker. As an example, consider a frame based character syntax in which the overall frame size is known, and for which one or more parameters are optionally included based on the value of other parameters. An attacker performing an exhaustive key search may find that, for a certain group of keys, the decrypted character indicates the inclusion of these optional parameters, and as a result the frame size is longer than it should be (this is called an overflow condition). Such a decrypted character is an "illegal" character that violates the character's syntax. In this case, the attacker can conclude that the correct decryption key is not in this group, which may save considerable time in completing the exhaustive search. Similar cases exist in underflow conditions, in which the decrypted syntax fails to use enough of the available bits in the frame to constitute a legal character. Finally, if one or more of the parameters in a character is precluded from taking on certain specific values, occurrence of these values can help an attacker rule out keys (this is called an illegal value condition and is another example of an illegal character that violates the character's syntax). A possible way to make the system more robust to attacks is to hash the data before it is encrypted. A hash function randomizes the data so that statistics and keywords cannot be used to help in an attack. Several problems exist with this method. It must be assumed that the attacker knows the hash function and can undo it. In addition, the hash function uses more processing power, further increasing the computational cost. Many data reduction systems exist that block process input data such as audio and video. These systems generally structure the compressed data in a predetermined fashion. The data is broken into blocks or frames that are independently decodable over some amount of time. The frames may consist of both fixed information (i.e., it is invariant from frame to frame) and variable information (i.e., it changes from frame to frame). Directly encrypting all of the data in these frames leads to the problems mentioned above, such as the large amount of processing required and the security problems with known keywords that could help an attacker determine the key. Known keywords contained within the fixed information such as sync words, data rate and other metadata change little or not at all from frame to frame and could be used to allow an attacker to devise an attack that would require less work than an exhaustive key search. It is known to encrypt a variable information portion of a frame prior to assembly of the output character by an encoder. However, this approach has several disadvantages. It is complex, requiring the encryption to be applied within the encoder rather than to the assembled character. This complexity renders it impractical for the case in which multiple copies of a character, each encrypted with a different encryption key, are to be sent to multiple users. In addition, if the encryption is applied to data within an encoder prior to entropy encoding [8], such as Huffman coding, the encryption tends to defeat the entropy coder's coding advantage (encryption tends to "whiten" the data, leaving fewer redundancies in the data for the entropy coding to reduce). Thus, there remains a need for an improved data encryption method.

II. Power Spectrum

The power spectrum of a signal is the power of that signal at each frequency that it contains. For example, white noise, which contains all frequencies at the same power, has a flat power spectrum. The power spectrum of the sound of the bottom key on a piano [4] would have a high value at the frequency corresponding to its note, and low values elsewhere.

III. Wav Signal Processing in MATLAB [1]

y = wavread(filename) loads a WAVE [10] file specified by the string filename, returning the sampled data in y. If filename does not include an extension, wavread appends .wav.

[y, Fs] = wavread(filename) returns the sample rate (Fs) in Hertz used to encode the data in the file.

[y, Fs, nbits] = wavread(filename) returns the number of bits per sample (nbits).

[y, Fs, nbits, opts] = wavread(filename) returns a structure opts of additional information contained in the WAV file. The content of this structure differs from file to file. Typical structure fields include opts.fmt (audio format information) and opts.info (text that describes the title, author, etc.).

[….] = wavread(filename, N) returns only the first N samples from each channel in the file.

[….] = wavread(filename, [N1 N2]) returns only samples N1 through N2 from each channel in the file.

[….] = wavread(…., fmt) specifies the data format of y used to represent samples read from the file. fmt can be either of the following values, or a partial match (case-insensitive):

'double'	Double-precision normalized samples (default).
'native'	Sample in the native data type found in the file.

siz = wavread(filename,'size') returns the size of the audio data contained in filename instead of the actual audio data, returning the vector siz = [samples channels].

IV. Fast Fourier Transform (FFT)

The Fast Fourier Transform does not refer to a new or different type of Fourier transform. It refers to a very efficient algorithm for computing the DFT. The time taken to evaluate a DFT on a computer depends principally on the number of multiplications involved. DFT needs N2 multiplications. FFT only needs Nlog2 (N).

The central insight which leads to this algorithm is the realization that a discrete Fourier transform of a sequence of N points can be written in terms of two discrete Fourier transforms of length N/2. Thus if N is a power of two, it is possible to recursively apply this decomposition until we are left with discrete Fourier transforms of single point.

V. Discrete Fourier Transform (DFT)

Using the Fourier series representation we have Discrete Fourier Transform (DFT) for finite length signal. DFT can convert time-domain discrete signal into frequency domain discrete spectrum. Fourier series links a continuous time signal into discrete-frequency domain. The periodicity of time-domain signal forces the spectrum to be discrete. The discrete Fourier transform of a discrete- time signal g[*n*] is given as

G[*k*]= $\sum_{n=0}^{N-1} g[n]\exp(-j2\pi nk/N)\,,\quad k = 0,1,2\,\dots.\,N-1$

Where, *N* is the number of time sequence values of g[*n*].It is also the total number of frequency sequence values in G[k], *T* is the time interval between two consecutive samples of the input sequence g[*n*], *F* is the frequency interval between two consecutive samples of output sequence G[k].

N, *T* and *F* are related by the expression

NT = 1 / F

NT is also equal to the record length. The time interval, T, between samples should be chozen between the Shanon's sampling theorem is satisfied. This means that should be less than the reciprocal of $2f_H$ *where* f_H is the highest significant frequency component in the continuous time signal g*[t]* from which the sequence g[*n*] was obtained. Several fast DFT algorithms require N to be an integer power of 2.

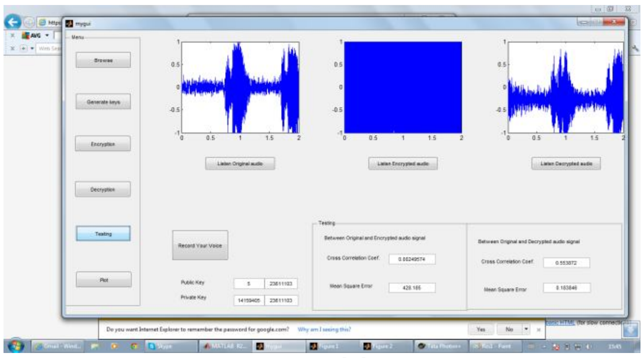
A discrete-time function will have a periodic spectrum. Both the time function and frequency functions are periodic in the DFT. Because the periodicity of DFT, it is common to regard points n=1 through n=N/2 as positive, and points from n=N/2 through n=N-1 as negative frequencies. In addition, DFT values at points n=N/2 through n=N-1 are equal to the DFT values at points n=N/2 through n=1 because both the time and frequency sequences are periodic.

An efficient method for computing the Discrete Fourier Transform is the Fast Fourier Transform (FFT).The number of computations needed for computing DFT can be reduced by FFT. For example, if a sequence has N points, and N is an integral power of 2, then DFT requires N^2 operations, whereas FFT requires $\frac{N}{2}\log_2 N$ complex multiplications, $\frac{N}{2}\log_2 N$ complex additions and $\frac{N}{2}\log_2 N$ complex subtractions. For N=1024 the computational reduction from DFT to FFT is more than 200 to 1.

The FFT can be used to obtain the power spectrum of a signal, do digital filtering, and obtain the correlation between two signals.

VI. OBSERVATIONS

The observation is taken on the recorded voice on our new software.


mygui

Menu: Browse, Generate keys, Encryption, Decryption, Testing, Plot

Listen Original audio | Listen Encrypted audio | Listen Decrypted audio

Record Your Voice

Public Key: 5 | 23611103
Private Key: 14159405 | 23611103

Testing

Between Original and Encrypted audio signal
Cross Correlation Coef. 0.00249574
Mean Square Error 428.165

Between Original and Decrypted audio signal
Cross Correlation Coef. 0.553872
Mean Square Error 0.183846

Do you want Internet Explorer to remember the password for google.com? Why am I seeing this? Yes No

Fig-1 Recorded Voices

The original wav signal is

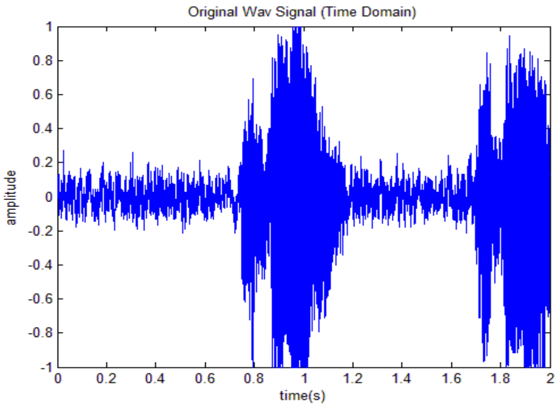


Fig-2 Original Wav Signal

The encrypted wav signal [1] is

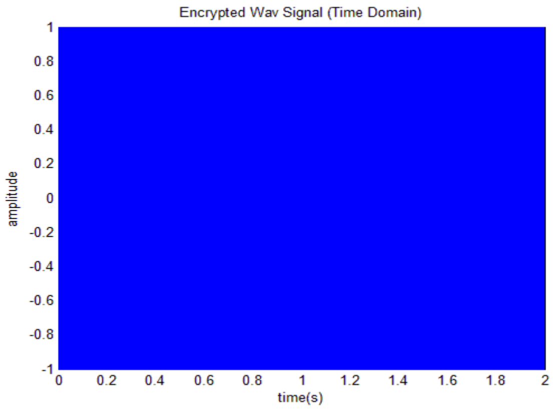


Fig-3 Encrypted Wav Signal

The decrypted wav signal is

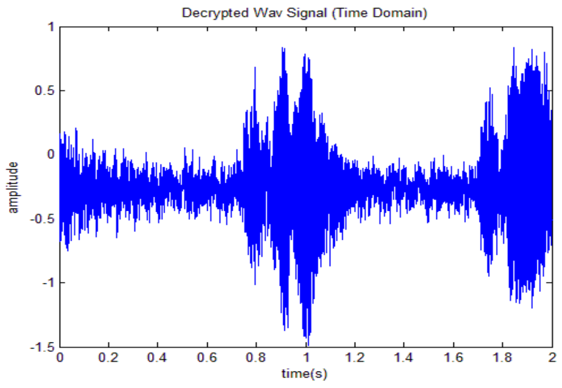


Fig-4 Decrypted Wav Signal

The original power spectrum in frequency domain is

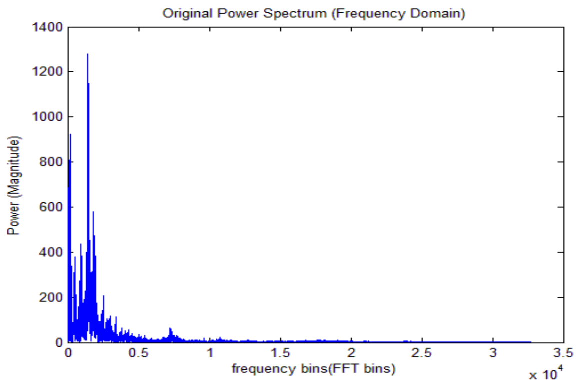


Fig-5 Original Power Spectrum

The encrypted power spectrum is

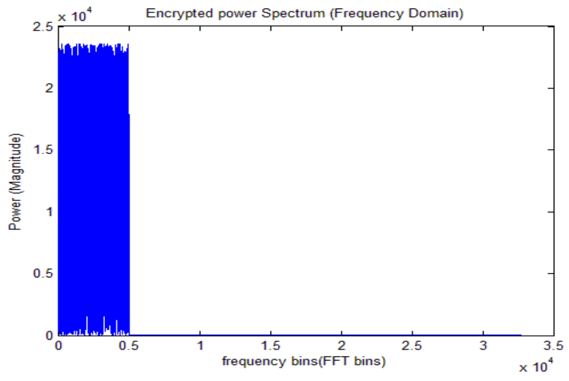


Fig-6 Encrypted Power Spectrum

The decrypted power spectrum is

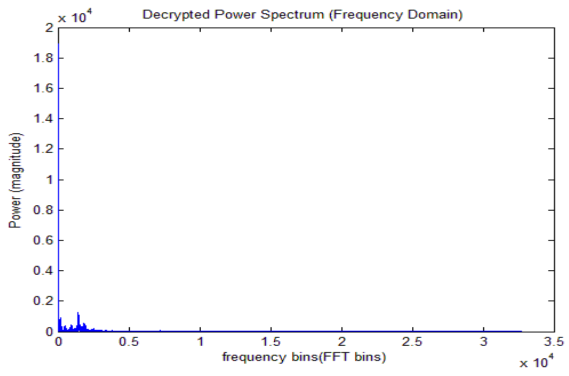


Fig-7 Decrypted Power Spectrum

VII. Conclusion and Future Work

From the above shown figures this is clear that decrypted wave form is close to the original wav form that we had while recording our audio i.e. we are receiving the same message as the sender send it to us. In this paper, we proposed a selective encryption approach. The proposed approach identifies and then encrypts important portions of the FFT coefficient (amplitude values).

The efforts that can be added to this proposed idea are developing an adaptive filter to analyze the useful audio sample dynamically.

Acknowledgment

I am thankful to my head of the department Mrs. Bhawna Mallick and co-ordinator Mr. Ajit Bharti whose continues guidance help me to complete this work. I am also thankful to my family and friends to allow and to work with me for the completion of this project.

REFERENCES

[1] www.mathworks.in/products/matlab/.

[2] Cryptography and Network Security Principles and Practices, Fourth Edition By William Stallings.C. Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, and M. Miller, “Rotation, scale, and translation resilient public watermarking for images,” IEEE Trans. Image Process., vol. 10, no. 5, pp. 767-782, May 2001.

[3] Rivest, R.; Shamir, A.; and Adleman, L. "A Method for Obtaining Digital Signatures and Public Key Cryptosystems." Communications of the ACM, February 1978.

[4] Agi, I. and Gong, L., “An Empirical Study of Secure MPEG Video Transmission,” Proceedings of the Internet Society Symposium on Network and Distributed System Security, San Diego, CA, February 1996, pp. 137-144.

[5] Qiao, L. and Nahrstedt, K., “A New Algorithm for MPEG Video Encryption,” Proceedings of the 1st International Conference on Imaging Science, Systems and Technology (CISST ’97), Las Vegas, NV, July 1997, pp. 21-29

[6] Wu, C.-P. and Kuo, C.-C. J., “Fast Encryption Methods for Audiovisual Data Confidentiality,” SPIE International Symposia on Information Technologies 2000, Boston, MA, November 2000, pp. 284-295

[7] Wu, C.-P. and Kuo, C.-C. J., “Efficient Multimedia Encryption via Entropy Codec Design,” *Proceedings of SPIE Security and Watermarking of Multimedia Content III*, Volume 4314, San Jose, CA, January 2001.10

[8] Zeng, W. and Lei, S., “Efficient Frequency Domain Selective Scrambling of Digital Video, ”IEEE Transactions on Multimedia, 2002

[9] In 2003 “Frequency –selective partial encryption of compressed audio” Servetti, A.; Testa, C.; De Martin, J.C.

[10] In May 2009 “Audio encryption using higher dimensional chaotic map” R. Gnanajeyaraman , K.Prasadh 2, Dr.Ramar3, Research scholar, Vinayaka Missions University, Salem, Tamilnadu, India.

[11] “Index-Based Selective Audio Encryption for Wireless Multimedia Sensor Networks”,H. Wang, Member, IEEE, M.Hempel, Member, IEEE, D.Peng, Member, IEEE, W. Wang, Member, IEEE, H. Sharif, Senior Member, IEEE, and H.-Hwa Chen, Fellow, IEEE,2010 IEEE.