Cryptanalysis of Keystream Reuse in Stream Ciphered Digitized Speech using HMM based ASR Techniques

L. A. Khan and M.S. Baig

Abstract— The keystream reuse problem in case of textual data has been the focus of cryptanalysts for quite some time now. This paper presents the use of hidden markov models based speech recognition approach to cryptanalysis of stream ciphered digitized speech in a keystream reuse situation. In this paper, we show that how an adversary can automatically recover the digitized speech signals encrypted under the same keystream. The technique is flexible enough to incorporate all modern speech coding schemes and all languages for which the speech recognition techniques exist. The technique is simple and efficient and can be practically employed with the existing HMM based probabilistic speech recognition techniques with some modification in the training (pre-computation) and/or the maximum likelihood decoding procedure. The simulation experiments, though preliminary, showed promising initial results by recognizing about 80 percent correct phoneme pairs encrypted by the same keystream.

***Index Terms*— cryptanalysis, hidden markov model, keystream reuse, speech recognition, stream cipher.**

I. INTRODUCTION

A stream cipher takes a plaintext p as input, exclusive OR it with a keystream k and produces ciphertext c as the output i.e. $p \oplus k = c$. If the keystream k is random and non repeating then the stream cipher becomes a perfect cipher [1] and is called the one time pad. The security of a stream cipher rests on never reusing the keystream. If two different plaintexts p_1 and p_2 are encrypted with the same keystream k then their results $p_1 \oplus k$ and $p_2 \oplus k$ can be XORed to neutralize the effect of the keystream k thereby obtaining $p_1 \oplus p_2$. The key reuse problem in stream ciphers and its exploitation in different scenarios have been studied since long. It has recently been mentioned in the literature as the "two time pad" problem [2]. The vulnerability of keystream reuse exists with many practical systems such as Microsoft Office [2, 3], 802.11 WEP [4], WinZip [5], PPTP [6] etc. This problem is predicted to remain there for quite some time in the future also [2]. There is a compelling need for a cipher mode of operation which can efficiently provide authenticated encryptions at speeds of 10 gigabits/s and is free of intellectual property restrictions. The counter mode of operation of a block cipher (e.g. AES) has been considered to be the best method for this purpose [23, 24], which has further increased the possibility of keystream reuse in actual systems [2].

Hidden Markov Models (HMMs) are very rich in mathematical structure and form the theoretical basis for use in a broad scope of applications, particularly in machine recognition of speech [7]. Most contemporary speech recognizers are based on HMMs. Digitization, compression and encryption of speech communications between two parties have been significantly important areas of communication. Encryption schemes particularly designed for speech, starting from the old aged analog speech inverters to the modern aged digital speech encryption techniques, has been the focus of security professionals since long. With the advancement in the speech digitization and compression techniques, the speech signal is now treated as an ordinary data stream of bits as far as encryption is concerned. But the acoustic and articulatory features of speech signals exploited by the automatic speech recognition (ASR) equipment especially in the distributed speech recognition (DSR) scenario [8] and automatic transcription of conversational speech [9] have encouraged us to look at their characteristics from the cryptanalytic point of view in a keystream reuse situation. We have extended the natural language approach from automated cryptanalysis of encrypted text based data to the digital data extracted from the underlying verbal conversation. An interesting by product of our attack is that it would not only decipher the information but would automatically transcribe it during the process of speech recognition.

The rest of the paper is organized as follows: In section 2, we discuss the prior work on the keystream reuse problem as well as the use of HMMs in cryptology. Section 3 presents our method of attack. In section 4 we present the implementation procedure which we adopted along with experimental results. Section 5, concludes the paper and gives directions for future work.

Manuscript received July 20, 2007.

The authors are with Center for Cyber Technology and Spectrum Management (CCT&SM), National University of Sciences and Technology (NUST), Pakistan. phone: +92-51-2103420/9266480.
(e-mail: liaquatalikhan@gmail.com, msbaig-cctsm@nust.edu.pk.)

II. Prior Work

A. Keystream Reuse Exploitation

Key stream reuse vulnerability exploitation of stream ciphers dates back to the National Security Agency's VENONA project [10,11] which started in 1943. Other worth mentioning works on the topic are that of Rubin, 1978 [12], Dawson and Neilson, 1996 [13] and the recent automated cryptanalysis of two time pads by Joshua Mason and coauthors in 2006 [2]. Mostly the keystream reuse problem discussed previously is with respect to the textual data and mainly based on heuristic rules for obtaining the two plaintexts p_1 and p_2 from $p_1 \oplus p_2$ except for [2] which uses statistical finite states language models and natural language approach. Prior works also exist on automated cryptanalysis of analog speech signals [13 ,14], but no previous work exists on the use of modern automated speech recognition (ASR) techniques based on hidden markov models (HMMs) being used for cryptanalysis of the two time pad problem for the digitally encoded and/or compressed speech signals.

B. Use of HMMs in Cryptology

As regards to the use of hidden markov models in cryptology, these have recently been used for several problems in this area. The most prominent are the works of A. Narayanan and V. Shamtikov who used hidden markov models for improving fast dictionary attacks on human memorable passwords [15]; D.X Song , D.Wagner and X. Tian who used HMMs for timing attacks on SSH [16]; substitution deciphering of compressed documents using HMMS by D. Lee [17]; L.Zhuang , F.Zhou and J.D works on keyboard acoustic emanations with the help of HMMs [18]; C. Karlof and D. Wagner who modeled countermeasures against side channel cryptanalysis as HMMs [19]; and finally the most relevant work of Joshua Mason and coauthors [2] who used the viterbi beam search for finding the most probable plaintext pairs from their XOR in case of textual data. It is worth mentioning here that most of the work involving cryptanalysis with the aid of HMMS relate to text based data with no or very little attention to digitized encrypted speech. Our algorithm for cryptanalysis of the plaintext XOR of the digitized speech signals using hidden markov model based speech recognition techniques is the first of its kind according to our knowledge and has showed encouraging preliminary results.

III. HMM Based ASR Techniques for Speech Cryptanalysis

Our method of cryptanalyzing the speech signals being encrypted with the same key is based on the hidden markov model based speech recognition techniques. The three basic questions with respect to the HMMs [7] and their solutions as regards to speech recognition are effectively utilized with some modification in our case. The three basic questions are:

1) Given an observation sequence *O* (XORed ciphered speech vectors in our case) and a model $\lambda = (\pi_i, A, B)$ where π_i is the initial probability of states, *A* is the transition probability of states and *B* is the emission probability distribution of the observation sequence, how do we compute the probability that the given sequence of observations was produced by the model λ i.e. $P(O/\lambda)$? The solution to this problem allows us to choose the model which best matches the observation sequence.

2) Given the observation sequence *O* of XORed speech vectors and the model λ , how do we choose a corresponding sequence of states i.e. XORed spoken words in case of isolated word recognizer and sequence of XORed phonemes in case of continuous speech recognizer, which is optimal and best explains the observation of XORed speech vectors? This is the problem in which we try to find the hidden part of the model i.e. to find the "correct" state sequence.

3) How do we adjust the model parameters $\lambda = (\pi_i, A, B)$ to maximize the probability of the observation sequences of XORed speech vectors given the model λ ? This is the training part of the model.

All the abovementioned problems and their efficient mathematical solutions have a very rich literature with respect to speech recognition [7]. In the conventional speech recognition techniques, the hidden markov models are trained for complete words in case of isolated word recognizers and for phones in case of continuous speech recognizers. In our case, for isolated word recognition, we have to first list down all the possible combination of words resulting from the XOR of the two speech signals and hence the HMMs required to be trained will increase from *n* to n^2. Since the list of words is generally very large, therefore, this approach of training the HMMs would be very computational intensive and maybe impractical. A better and more efficient approach, which is also used in large vocabulary continuous speech recognition (LVCSR), is to train the HMMs for the exclusive ORed pairs of all the possible phonemes in the language under test. For example in English language there are about *40* to *50* phonemes and hence the number of HMMs to be trained in this case would be at the most 50^2 which is not high as regards to the computational resources available to a normal user these days. Using this approach would not require any modification in the conventional speech recognition procedure, except that the decoded phonemes would not be the actual speech signals but the bitwise XORed of two phonemes. In this case it is only in the training phase that, instead of training the HMMs for individual phonemes, we train them for the XORed pairs of phonemes for all combinations.

IV. Implementation

The simulation part of our attack involves a pre computation phase and then the actual attack phase. Both the phases are interrelated and interdependent and the accuracy of the attack is greatly dependent on how well these two parts of the attack are carefully employed and joined. For both these phases, we used

HTK which is an open source toolkit [20] based on *C* language available for building hidden markov models. It is primarily designed for building HMM based speech processing tools, particularly speech recognizers. *HTK* training tools are used to estimate a set of HMMs using training utterances and their associated transcriptions whereas unknown utterances are transcribed using the *HTK* recognition tools. Both these tools can be efficiently employed to break the two time pads of ciphered digitized speech.

A. Pre-computation Phase

The pre computation phase corresponds to the training part of the HMMs and is done once for a particular language and specific speech encoding procedure. In order to prove the concept, we present a simple example in which we take ten different utterances by different speakers of the two phonetically balanced English sentences: *Clothes and lodging are free to new men*; and *All that glitters is not gold at all*. We bit wise XORed the digital encoded forms of these sentences to simulate the keystream reuse scenario. Fig. 1(a), (b) show the spectrogram and waveform of the two sentences along with their transcription. The transcription at the phoneme level is obtained from the British English pronunciation dictionary BEEP [22]. For simplicity of implementation the silence between words is not marked separately, only the initial silence and the end silence are marked. Fig 1(c) corresponds to the bitwise XOR of the two signals and the associated transcriptions at the phoneme level. For the individual sentences the number of phonemes is 27 for sentence 1 and 26 including silence (*sil*) and hence the HMMs required to be trained for the XOR case would be 702 at the max. These are obtained by pairing every phoneme of sentence 1 with every phoneme of sentence 2. We used ten utterances each of the sentence 1 and sentence 2 from ten different speakers, labeled the *wave* files and then bit wise XORed both the files again labeling these with the HMM boundaries clearly defined. These recordings and transcription can be obtained by the *HTK* tool *HSLab*. The above mentioned acoustical events were modeled by 167 HMMs with each HMM corresponding to one XORed pair of phonemes. Since all the possible phonemes do not occur hence the actual number (167) of phoneme pairs is quite less than the total possible number (702). The basic design of the HMM we used in this case for all the models is as shown in Fig. 2. The configuration we used for the speech recognition was based on Mel Frequency Cepstral Coefficient (MFCC) [21] with 12 first MFCC coefficients, the null MFCC coefficient which is proportional to the total energy in the frame, 13 Delta coefficients estimating the first order derivative of MFCC coefficients and 13 acceleration coefficients estimating the second order derivatives, altogether a 39 coefficient vector is extracted from each signal frame. The frame length is 25 milliseconds with 10 milliseconds frame periodicity. The parameters which are to be estimated for each HMM during the training phase are transitional probabilities a_{ij} and the single Gaussian observation function for each emitting state which is described by a mean vector and variance vector (the diagonal elements of the autocorrelation matrix). In our case we have to estimate all these values for each of the 167 HMMs during the training phase. The *HTK* tools *HInit*, *HCompV*, and *HRest* can be used for this purpose.

Before using our HMMs we have to define the basic architecture of our recognizer. In actual case this depends on the language and the syntactic rules of the underlying task for which the recognizer is used. We assume that these things like the language of the speakers and the digital encoding procedures are known to the cryptanalyst before hand. *HTK*, like most speech recognizers, works on the concept of recognition network which are to be prepared in advance, and the performance of the recognizer is greatly dependent on how well the recognition network maps the actual task of recognition. In addition to the recognition network, we need to have a task dictionary which explains how the recognizer has to respond once a particular HMM is identified. The task grammar for our recognition network is shown in Fig. 3. The recognition network for our experiment is shown in Fig. 4. The *HParse* tool of *HTK* can be used for this purpose. *HSGen* can be used for its verification.

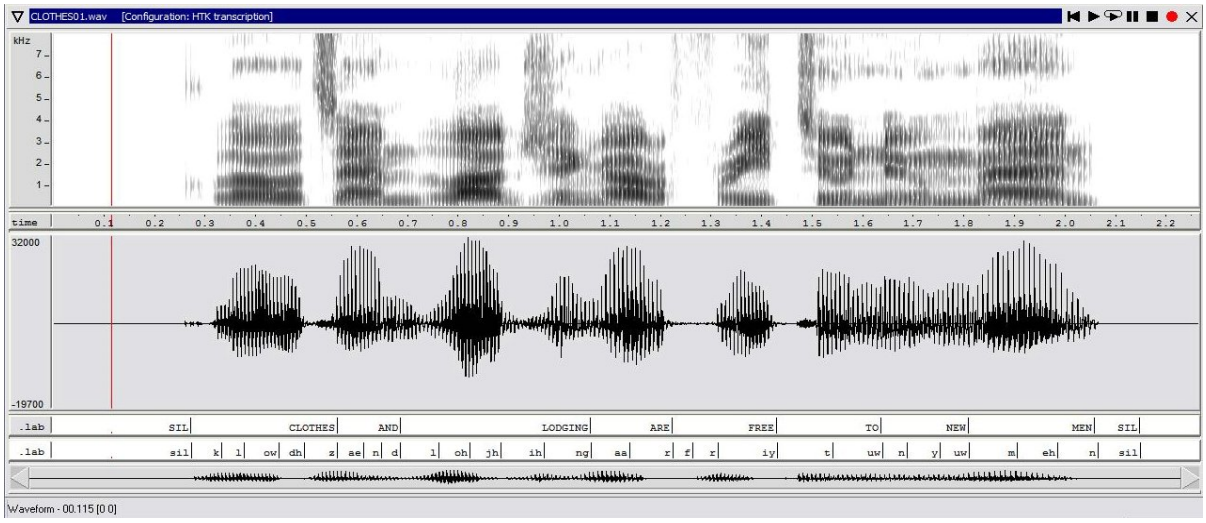

Fig. 1(a). Spectrogram and waveform along with transcription of the sentence:

Clothes and lodging are free to new men.

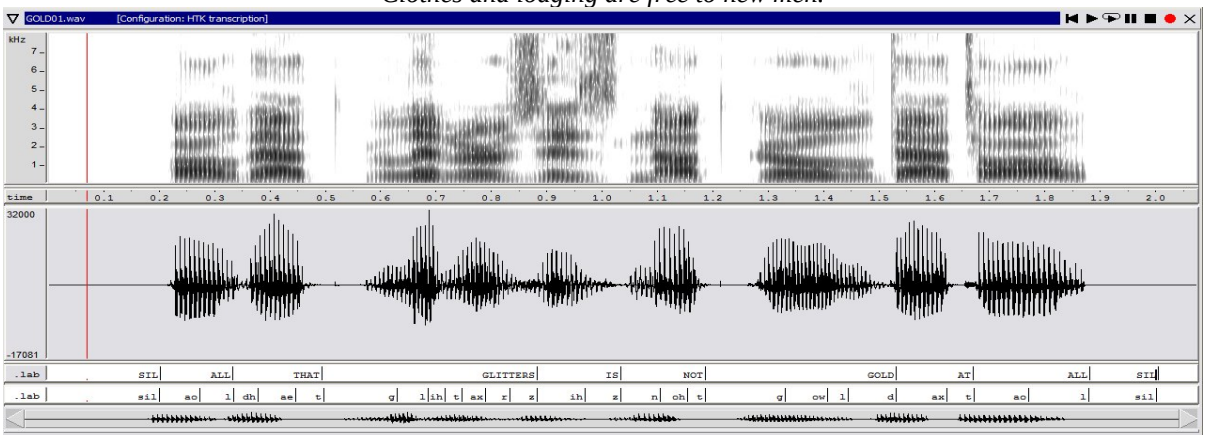

Fig. 1(b). Spectrogram and waveform along with transcription of the sentence:
All that glitters is not gold at all.

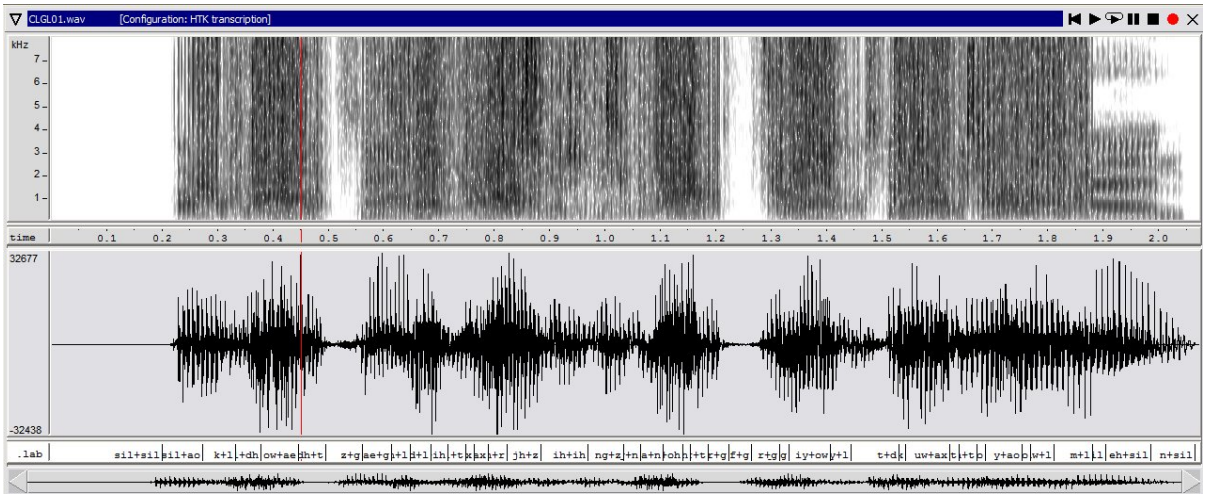

Fig. 1(c). Spectrogram and waveform along with transcription of XOR of the sentences.

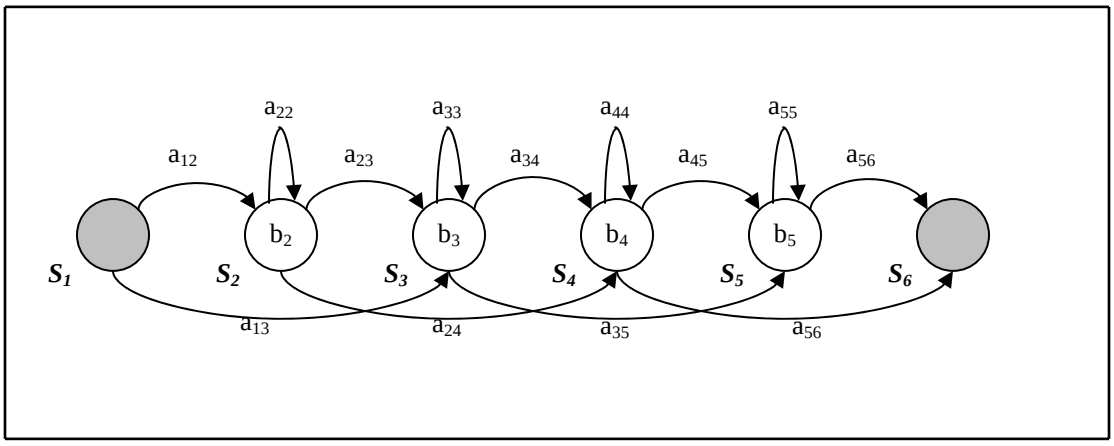

Fig. 2. Basic Topology of the HMM

```
/* Task grammar*/
$WORD = sil+ao | k+l | l+dh | ow+ae | dh+t | z+g | ae+g | n+l  | d+l | l+ih | l+t | l+ax | oh+ax | oh+r | jh+z |ih+ih
| ng+z | ng+n | aa+n | r+oh | r+t | r+g | f+g | iy+g | iy+ow | iy+l | t+d | t+ax | uw+ax | uw+t | n+t | n+ao | y+ao |
uw+ao | m+l | m+sil | eh+sil | n+sil | k+ao | ow+dh | ow+t | dh+g | z+l | ae+l | ae+ih | n+ih | d+t | l+r | oh+z |
jh+ih | ih+z | aa+t | f+ow | r+ow | r+l | iy+d | iy+ax | iy+t | t+t | y+l | uw+sil | sil+l | f+oh | f+t | n+ax | y+t | m+ao
| eh+l | l+ao | ow+ao | dh+ao | ae+dh | ae+ae | d+g | l+g | oh+l | jh+l | ng+t | aa+ax | r+ax | r+r | f+r | f+z | f+ih |
r+ih | iy+z | t+z | uw+n | n+oh | uw+g | uw+ow | m+ow | eh+ow | n+d |  sil+ax | sil+t | n+ae | d+ae | l+ae | oh+ae
| oh+t | oh+g | jh+g | ih+l | ng+l | aa+ih | f+ax | r+z | iy+ih | uw+z | n+n | y+oh | m+g | eh+g | n+ow | sil+ow |
sil+d | l+l | ow+l | dh+l | z+dh | jh+t | ih+t | ng+g | aa+g | f+l | n+z | y+z | m+z | eh+n | eh+oh | sil+oh | dh+dh |
ih+g | aa+l | t+ih | uw+ih | m+n | sil+g | k+sil | n+dh | ng+ax | aa+r | aa+z | f+n | l+sil | ow+sil | z+ao | ih+ax |
ng+r | r+n | iy+oh | r+d | ae+t | oh+ih | jh+ax | ih+r | t+g | y+d | uw+d | m+ax | eh+ax | aa+oh ;

( [ START_SIL ] { $WORD } [ END_SIL ] )
```

Fig. 3. Task Grammar for the Recognition Network

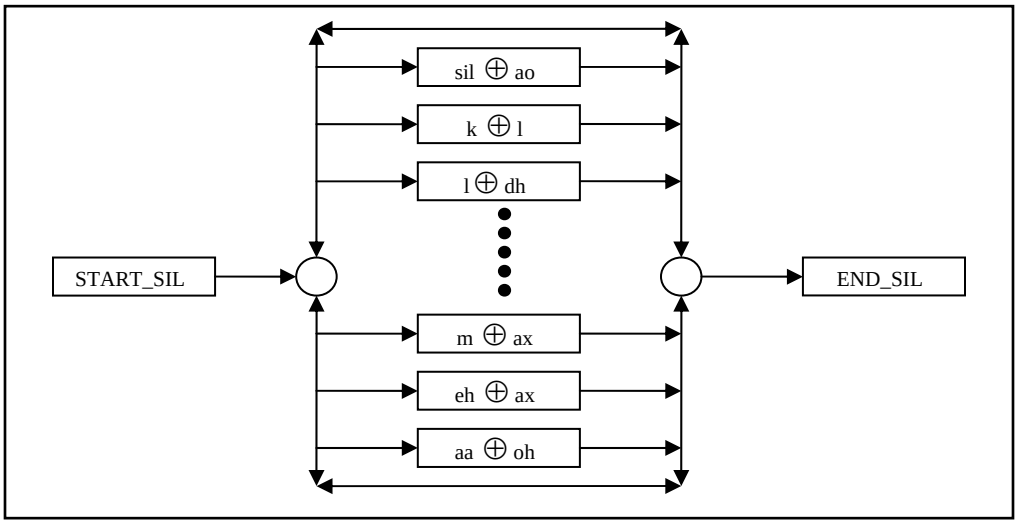

Fig. 4. Recognition Network

A. Decoding Phase

Once the pre-computation phase has carefully been completed, the decoding process becomes pretty simple and elegant. An input speech signal comprising of *n* observation vectors, which in our case are the XOR of two unknown sequences of vectors, is then fed as input to the recognizer. Every path from the start node to the end node in the recognition which passes through precisely *n* emitting states is a prospective recognition hypothesis. Each of these paths has a log probability which is computed by summing the log probability of individual transition in the path and the log probability of each emitting state generating the corresponding XORed vector. Within the model, transitions are determined from the model parameters (a_{ij}), between two models the transitions are regarded as constant and in case of large recognition networks the transition between end words are determined by language models likelihoods attached to the word level networks [20]. The decoder lists those paths through the network which have the highest log probability. These paths are found using a *Token Passing Algorithm* [20]. At time 0, a token is placed in every possible start node. Each time step, tokens are propagated along connected paths through the recognition network stopping whenever they hit an emitting HMM state. When there are more than one out going paths from a node, the token is copied so that all possible paths are explored in parallel. As the token passes across transitions the corresponding transition and emission probabilities add up to its log probability. The token also maintains a history of its route during the process of propagation. In a large network, which will definitely be in our the case, a *beam search* may be used in case of which a record of the *best token* overall is kept while deactivating all tokens whose log probability falls more than a *beam width* below the best. This *beam search* technique has one problem i.e. if the pruning beam width is set too small then the actual recognition path might be pruned before its token reaches the end of the observation i.e. it may result in a *search error*. Setting the beam width is thus a compromise between computational load and avoiding *search errors*. Fortunately, *HTK* tools *HVite* takes care of all these speed and computation problems [20].

B. *Experimental Results*

The performance analysis of the recognizer can be done using the *HResults* tool of *HTK*. It reads in a set of label files output by the recognition tool (*HVite* in our case) and compares them with the corresponding reference transcription files. For the analysis of speech recognition output the comparison is based on a *dynamic programming based string alignment procedure* [20]. The experimental results of the recognition of the XORed phonemes obtained by *HResults* is depicted in Fig. 4. The first line in the overall results gives the sentence level accuracy based on the total number of labels identical to the transcription. The sentence level accuracy is quite low i.e. one out of the 10 sentences has been identified completely correct. It does not give realistic picture of the recognizer performance because if even one of the phoneme pair is incorrect the whole sentence is considered incorrect. Therefore, the world level correct percentage gives a realistic performance analysis of the recognizer which is more than 85% in our case. The second line depicts the word level accuracy in which H corresponds to the number of correct labels, D is the number of deletions, S is the number of substitutions, I is the number of insertions and N is the total number of labels in the defining transcription files.

```
=================== HTK Results Analysis==============

Date: Mon Jun 11 20:05:23 2007

Ref : data2/ref22.mlf

Rec : data2/rec22.mlf

-------------------------Overall Results ---------------

SENT: %Correct=10.00 [H=1, S=9, N=10]

WORD: %Corr=86.98, Acc=83.72 [H=374, D=10, S=46, I=14, N=430]

====================================================
```

Fig 5. HTK Recognition Performance

V. Conclusions and Future Work

In this paper, we presented that how the keystream reuse problem of stream ciphers can be exploited in case of speech signals encoded with modern encoding techniques. The experimental results, though very preliminary, have showed promising results, thereby, paving the way for further research on the topic. The details and complexity analysis of modifying the decoding phase of the recognizer can be taken up as a future work. *HTK* and other automatic speech recognition (ASR) tools work on the concept of context dependent tied state multi mixture triphones [20] which make the performance of the recognizer more flexible and robust. The use of triphones in the keystream reuse scenario needs to be looked into in the future assignments. The performance of the recognition system in the case of eavesdropped encrypted telephone conversation needs to be evaluated which could be a direct security application of our approach as the *HTK* recognizer has already shown encouraging performance in the transcription of telephone bandwidth speech communication [9].

References

[1] C.E. Shannon. *A mathematical theory of communication*. Bell System Technical Journal, 27:379-423, July, 1948.

[2] Joshua Mason, Kathryn Watkins, Jason Eisner and Adam Stubblefield. *A natural language approach to automated cryptanalysis of two time pads.* In 13th ACM Conference on Computer and Communications Security, Nov, 2006.

[3] H. Wu. *The misuse of RC4 in Microsoft Word and Excel*, Cryptology ePrint Archive, Report 2005/007, 2005. http://eprint.iacr.org.

[4] N. Borisov, I.Goldberg and D. Wagner. *Intercepting mobile communications: The insecurity of 802.11*. In MOBICOM 2001, 2001.

[5] T. Kohno. *Attacking and repairing the WinZip encryption scheme*, In 11th ACM Conference on computer and communications security, pp 72-81, Oct 2004.

[6] B. Schneier, Mudge and D. Wagner. *Cryptanalysis of Microsoft PPTP Authentication Extensions (ms-chapv2)*. CQRE'99, 1999.

[7] L.R. Rabiner. *A tutorial on hidden markov models and selected applications in speech recognition*, Proceedings of the IEEE, 77(2): 257-286, Feb, 1989.

[8] David Pearce. *Enabling new speech driven services for mobile devices: An overview of the ETSI standards activities for distributed speech recognition front ends*. AVIOS 2000: The Speech Applications Conference, CA, USA, May, 2000.

[9] M. J. F. Gales, B. Jia, X. Liu, K. C. Sim, P. C. Woodland and K. Yu. *Development of the CUHTK 2004 RT04F Mandarin Conversational Telephone Speech Transcription System. Proc. ICASSP 2005, Volume I, pp. 841-844, March 2005.*

[10] P. Wright. *Spy Catcher*. Viking, New York, NY,1987.

[11] R. L. Benson and M. Warner. *VENONA: Soviet Espionage and the American Response 1939-1957*. Central Intelligence Agency, Washington D.C., 1996.

[12] R. Rubin. *Computer methods for decrypting random stream ciphers*. Cryptologia, 2(3):215-231, July 1978.

[13] E. Dawson and L. Nielsen. *Automated cryptanalysis of XOR plaintext strings*. Cryptologia, 20(2): 165-181, April, 1996.

[14] B. Goldburg, E. Dawson, S. Sridharan. *The automated cryptanalysis of analog speech scramblers*, Adnances in Cryptology, EUROCRYPT'91, Springer-Verlag LNCS 457, pp 422, April, 1991.

[15] A. Narayan and V. Shmatikov. *Fast dictionary attacks on human-memorable passwords using time-space trade-off*. 12th ACM Conference on Computer and Communications Security, pp 364-372, Washington D.C., Nov, 2005.

[16] D. X. Song, D. Wagner and X. Tian. *Timing analysis of keystrokes and timing attack on SSH*. 10th USENIX Sec. Symposium, Aug, 2001.

[17] D. Lee. *Substitution deciphering based on HMMs with application to compressed document processing*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(12): 1661-1666, Dec, 2002.

[18] L. Zhuang, F. Zhou and J. D. Tygar. *Keyboard acoustic emanations revisited*. 12th ACM Conference on Computer and Communications Security, pp 373-382, Washington, D.C., Nov, 2005.

[19] C. Karlof and D. Wagner. *Hidden markov models cryptanalysis*. Cryptographic Hardware and Embedded Systems- CHES'03, Springer-Verlag LNCS 2779, 17-34, 2003.

[20] *HTK*- Hidden Markov Model Toolkit – Speech Recognition Toolkit and Documentation. http://htk.eng.cam.ac.uk/download.shtml.

[21] V. Tyagi and C. Wellekens, *On desensitizing the mel-cepstrum to spurious spectral components for robust speech recognition.* ICASSP '05. vol. 1, 2005, pp. 529–532.

[22] BEEP- British English Pronounciation Dictionary (Phonetic Transcriptions of over 250,000 English words). http://svr-www.eng.cam.ac.uk/comp.speech/Section1/Lexical/beep.html.

[23] David A. McGrew and John Viega, The Galois/Counter mode of Operation (GCM) , May, 2005. Available from http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-revised-spec.pdf.

[24] R. Housley and A. Corry, GigaBeam high speed radio link encryption, RFC 4705, Oct, 2006. Available from http://tools.ietf.org/html/rfc4705.