ANALYZING ENCRYPTED SPEECH

By

Liaqat Ali Khan

A thesis submitted to the Faculty of Information Security Department,

Military College of Signals,

National University of Science and Technology, Pakistan

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Information Security

May 8, 2010

ABSTRACT

Speech is digitized, encrypted and sent between two parties in many situations. Digitized speech signals are generally considered as ordinary binary data streams as far as encryption is concerned. In this thesis, we present that the properties of speech signals are different from text, image, video and other non speech signals and need special attention while being encrypted. These properties of speech signals have to be kept in mind, not only, while designing encryption algorithms for digitized speech signals but also during implementation of these algorithms in software and hardware. We present that how the statistical properties of speech can be utilized to extract important information, from cryptanalytic point of view, from the encrypted speech signals. We also study some of the published cryptanalysis techniques particulary used for text based data and then study the effectiveness of these techniques if the underlying plaintext data is digitized and compressed speech. We discuss a particular situation in stream ciphers called the two time pad situation, assuming the two underlying signals being speech. The two time pad situation is presented at length in the literature with respect to text based data and not discussed at all with respect to secure speech communications. We present techniques for exploitation of two time pad situation in case of stream ciphered digitized speech. Separate techniques for exploitation are presented for waveform (uncompressed) and parameter encoded (compressed) speech. We also analyze the latest techniques of selective encryption applied on compressed speech signals and present that some of these techniques can be effectively employed for content protection but may not conceal the speaker's identity. In the last part of the thesis we analyze the latest encryption techniques used in voice over IP applications. We discuss that if speech is compressed using variable bit rate encoding resulting into variable length packets can be used for speakers recognition even after encryption. Experimental evaluation results for all the above techniques are presented.

DEDICATION

This work is dedicated to my family who have always been there to help me and without whose patience and support, it would have not been possible.

ACKNOWLEDGMENT

I acknowledge that this thesis would have not been possible without the ongoing support and guidance of my thesis supervisor Dr. Muhammad Shamim Baig. He has been always there to answer my silly questions and has been patiently listening to my long and boring discussions for hours and hours. I also acknowledge the help and support of my felow students and colleagues at Center for Cyber Technology and Spectrum Management(CCT&SM) who have provided me an excellent atmosphere during my four years stay at CCT&SM. I would especially like to mention the names of Ali Hassan, Samin Khaliq, Tariq Saeed and Muhammad Tariq who have always supported me in every thick and thin. Mr. Ali Hassan has been kind enough to even code my simulations and experiments in MATLAB whereas Miss Samin has always been there to help in C language programming and parallel implementations. Tariq Saeed and Muhammad Tariq has not only helped me in proof reading of my work and but has always been encouraging me to complete my doctoral research. They have been entertaining me with their lively discussion and humorous tales even in the most stressful phases of my research work and thesis writing. I would also like to mention the name of Dr. Fauzan Mirza, who in spite of not being in any of my committee or institute, has been providing me his able guidance with a very pleasant and soft spoken attitude during the research phase. I would be highly unjust not to mention the names of my fellow PhD students Nazir Malik, Firdous Kausar and Mehreen Afzal. Nazir Malik has been there to share with me the most stressful part of the qualifying exam. I admit that without his encouragement and support I would have not been able to qualify that. The untiring and continuous support of Mehreen Afzal throughout the research phase of my PhD and the long and arduous discussion between us have been the source of support and inspiration for me. I owe all my publications to her who has always been there to keep me informed of the relevant conferences and journals of our area of research. I would also like to mention the name of Dr. Amr Yousssef with whom I worked as a visiting researcher during my stay at Concordia Institute of Information System Engineering(CIISE). It was due to his able guidance and unprecedented support which helped me in producing some significant contributions in the area of speaker recognition from encrypted speech. I owe the second part of my thesis to him and the team at CIISE including Farkhund Iqbal, Esam Elsheh, Mohammad Rasslan, Hamad Bin Saleeh and Abdal Aleem.

TABLE OF CONTENTS

LIST OF FIGURES

LSIT OF TABLES

Chapter 1

INTRODUCTION

1.1 Speech

Speech is one of the basic and direct form of human interaction with each other. The syntactic combination of lexicals and names drawn from very large vocabularies form the basis of this human interaction. Each spoken word is created from the phonetic combination of a limited set of vowel and consonant speech sound units. In the literature speech is broadly studied from two different perspectives i.e. *speech production* and *speech perception*. Several academic disciplines study speech production and perception which include acoustics, psychology, speech pathology, linguistics, cognitive science, communication studies and computer science. We, in this thesis, discuss speech from the perspective of a branch of computer science i.e. from information security point of view. We study speech while being sent from one person to another via some communication medium in a secure form. In this, we study the peculiar properties of speech and try to exploit these properties in order to gain some knowledge about the underlying security mechanism, the speaker's identity or the plain speech signal.

1.1.1 Speech Production

Most of the speech processing techniques currently in use greatly are either directly derived from or at least strongly influenced by human speech production system and its principles [1]. The basic human speech production system is depicted in Fig. 1.1. Air is expelled by the lungs upwards

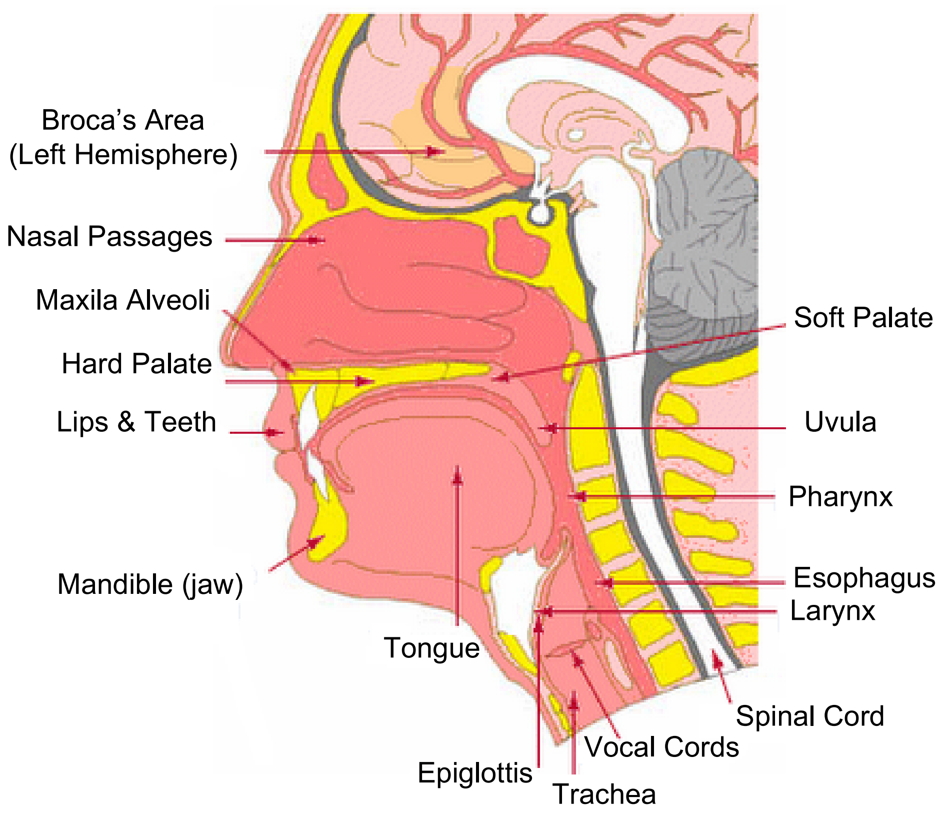

Figure 1.1: Human Speech Production System

through the trachea which then passes through the larynx. When the air passes through the tensed vocal cords in the larynx they vibrate. As a result the air adopts a roughly periodic flow which is then modulated in frequency while passing through the throat, the mouth cavity, and the nasal cavity. The sound created is mainly radiated from the lips and, to some extent, from the nose. The pharyngeal and mouth cavities in combination forms the vocal tract. The portion between the velum and the nostrils forms the nasal tract. Certain nasal sounds are produced when the velum acts as a trap door to acoustically couple the nasal tract.

This design of human speech production system has been the basis of most of the speech processing applications such as speech coding, speech compression and speech synthesis. An engineering simulation of the human speech production system is presented in Fig. 1.2.

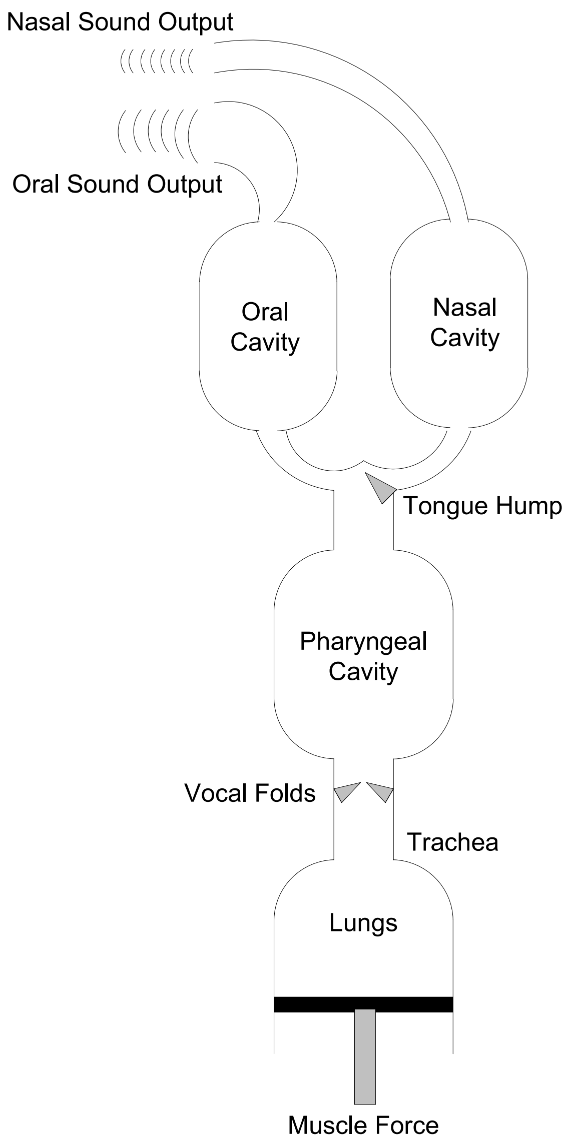

Figure 1.2: Engineering Simulation of the Human Speech Production System

The throat , mouth and nasal cavities combinely act as an acoustic filter. The properties of this acoustic filter are mainly determined by the changing position of the tongue, jaw, lips, velum, mouth, etc. which are known as the articulators. The shape of the vocal tract is characterized by a set of resonant frequencies called formants. The formant central frequencies are usually denoted by $F_1, F_2, F_3, \cdots$ with lower numbers corresponding to lower frequencies. The overall spectrum of the voice is mainly shaped by the first 35 formant frequencies by amplifying certain frequencies while attenuating others. The glottis is an opening between the vocal folds. Voiced sounds like the sound associated with vowels in English language are produced when the vocal cords are tensed and vibrate and the air passing through the glottis becomes quasi-periodic. For men, the pitch range for voiced sounds generally varies between 50-250 Hz, while the range is approximately 120-500 Hz for women [1]. When the vocal cords are relaxed, sound can still be produced. Unvoiced sounds are produced when the airflow passes through a restriction in the vocal tract and become turbulent. An example of unvoiced sound is the sound associated with *s* in *six*). Plosive sounds (e.g. the sound associated with the letter *t* in English) is produced when there is a point of total closure in the vocal tract (usually near the front) so that the air pressure builds up behind this point and when the closure is opened, the air is suddenly released causing a brief transient sound. Mixed sounds can also be produced which are simultaneously voiced and unvoiced. An example of such sound is *s* in *his*, pronounced like a *hiz*).

1.1.2 Speech Perception

Speech perception is the process by which humans are able to understand and interpret the sounds used in a language. Speech perception is closely associated with the fields of phonetics and phonology in linguistics and cognitive psychology and perception in psychology. It seeks to understand how a human listener recognizes speech sounds and use this information to understand spoken language. The research in this field of Speech involves applications in building computer systems for speech recognition, as well as improving speech recognition for hearing- and language-impaired listeners. The design of efficient speech synthesis, recognition and even encoding, compression and encryption systems for human speech signals are mainly derived by the human speech perception. A block level description of the human speech production and perception is depicted in Fig. 1.3

Apart from the the above two main speech research directions, following are some of the important speech processing techniques which are researched, discussed and applied in direct or indirect human speech interactions especially after the development of digital processing techniques:-

1. Speech Encoding and Compression
2. Speech Synthesis
3. Speech Recognition
4. Speech Encryption

1.1.3 Speech Encoding and Compression

Speech coding is different from all other forms of audio coding. Speech is a much simpler signal than most other audio signals. Moreover, a lot of statistical information is available prehand about the properties of speech [2]. As a result, some auditory information which is relevant in other forms of audio coding may be completely or partially irrelevant in the speech coding context. In speech coding, the most important criterion is preservation of intelligibility and quality of speech, with a limited amount of transmitted data. The intelligibility of speech, in addition to the actual literal content also includes the speaker's identity. In addition to his identity his state

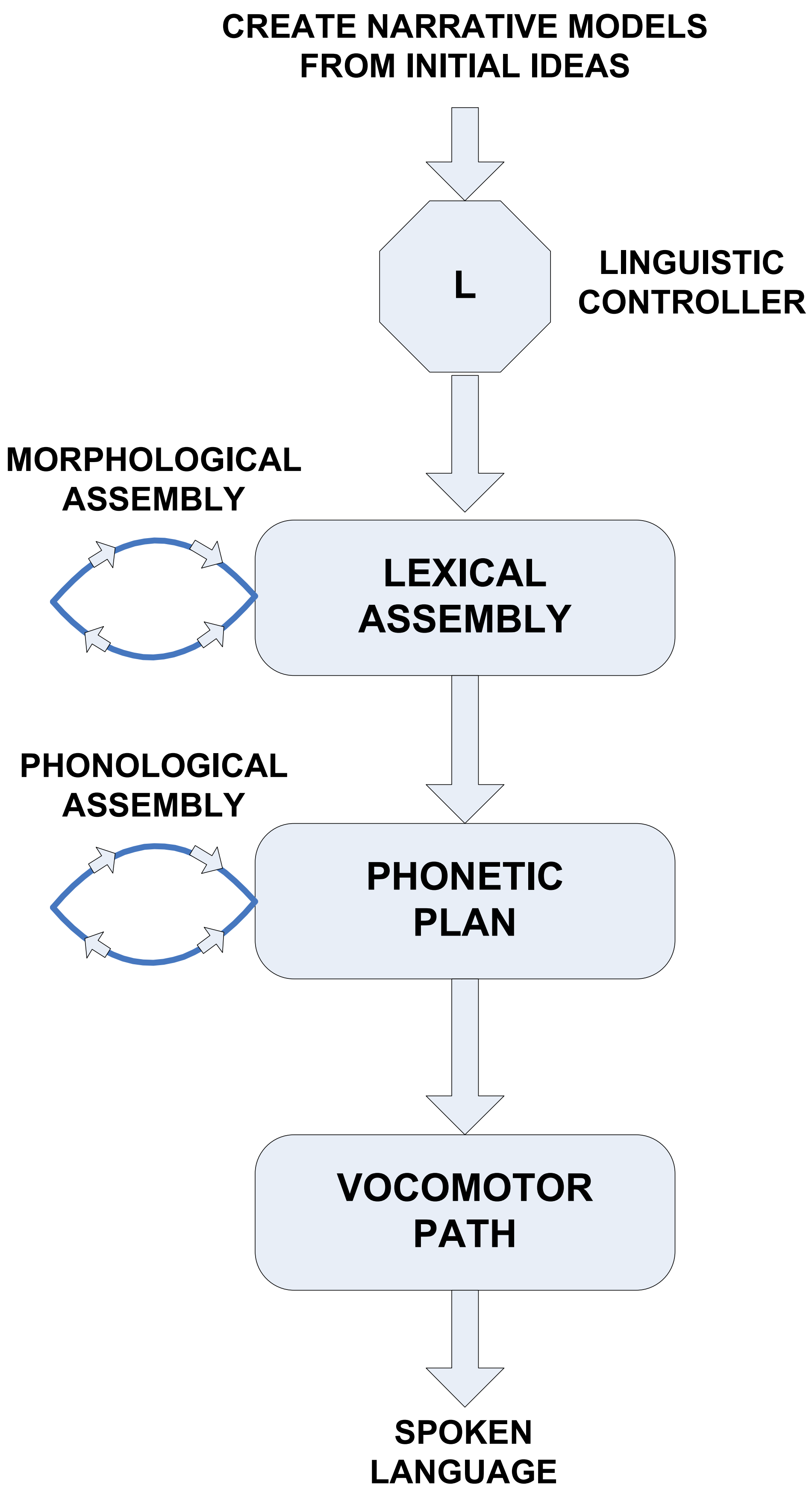

Figure 1.3: Block Level Simulation of the Human Speech Production System

of mind, emotions, intonation, timbre etc are also important. The concept of quality of speech is more synonymous to pleasantness. It is completely different from intelligibility, since it is possible that degraded speech is completely intelligible, but may be qualitatively annoying to the listener. Another important factor which has to be kept in mind while designing speech processing/transmisiion applications is that most speech applications require low coding delay, as long coding delays may degrade the speech quality and interfere with speech interaction. In the speech coding domain, the A-law and -law algorithms (G.711) [3] used in traditional Pulse Coded Modulated (PCM) digital telephony can be seen as a very early precursor of speech encoding. These encoding techniques require only 8 bits per sample but gives effectively a resolution of 12 bits. This type of coding mechanisms, if employed over music signals would generate unacceptable distortion. Speech Waveforms have a peculiar peaky nature and simple frequency structure. It has a periodic waveform with a single fundamental frequency and occasionally added noise bursts. It is because of this peculiar nature of speech signals that the very simple instantaneous compression algorithms of A-law and -law are perfectly suitable for coding. At the time of their design, their low complexity with good (33%) bandwidth reduction made these techniques excellent coding mechanisms. Much of the later work in speech compression was motivated by military research into digital communications for secure military radios, where very low data rates were required to allow effective operation in a hostile radio environment [2]. Today, a lot of processing power is available today as compared to the one available for earlier compression techniques. As a result, modern speech compression algorithms could use far more complex techniques thereby achieving much higher compression ratios.These techniques were available through the open research literature to be used for civilian applications, allowing the creation of digital mobile phone networks with substantially higher channel capacities than the analog systems that preceded them. In addition to the actual speech coding of the signal, it is often necessary to use channel coding for transmission, to avoid losses due to transmission errors. Usually, speech coding and channel coding methods have to be chosen in pairs, with the more important bits in the speech data stream protected by more robust channel coding, in order to get the best overall coding results. The encoding and compression mechanisms for speech are developed based on the following basic properties of speech signal [4].

- Speech signal maintains a particular frequency, the peaks generally occur after 65-85 samples with sampling rate of 8000 samples per second.
- Speech signal has smooth ups and downs and very seldom does abrupt changes take place in the amplitude of the signal.
- Speech signal never continues to increase or decrease, it generally to go down after some time as much as it has increased.
- Speech signal contains noise, after being smoothed by averaging operator, the noise lie between two lines parallel to x-axis.
- Speech signal generally maintains an amplitude.

Speech coding and compression applications can be divided into two broad categories, narrow band speech coding and wide band speech coding.

1.1.3.1 Narrow Band Speech Coding

This type of encoding considers speech signal between 300 to 3400 Hz of frequency band. The biggest application of this type of encoding mechanism is the telephone network. Other applications include Narrow-band speech coding FNBDT (Future Narrow Band Digital Terminal) [5] for military applications, SMV (Selectable Mode Vocoder) for CDMA networks [6], Full Rate, Half Rate, EFR, AMR for GSM networks [7], G.723.1, G.726, G.728, G.729 [8], Internet Low Bit Rate Codec (iLBC) [9] and others for voice over IP or video conferencing applications.

1.1.3.2 Wide Band Speech Coding

This type of encoding takes as input the speech signal in the frequency band of 50 to 7000 Hz. Typical applications in this range include Wide-band speech coding AMR-WB for WCDMA networks, VMR-WB for CDMA2000 networks, G.722, G.722.1, Speex and others for VoIP and video conferencing applications.

1.1.4 Speech Synthesis

Speech synthesis is the artificial production of human speech. A system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware. Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database. Systems differ in the size of the stored speech units; a system that stores phones or diphones provides the largest output range, but may lack clarity. For specific usage domains, the storage of entire words or sentences allows for high-quality output. Alternatively, a synthesizer can incorporate a model of the vocal tract and other human voice characteristics to create a completely ”synthetic” voice output [2]. The quality of any speech synthesis system is judged by how similar it produced sound is to the human voice, and by its ability to be understood. An intelligible speech synthesis system helps people with visual impairments or reading disabilities to listen to written works. Many computer operating systems have included speech synthesizers since the early 1980s. The most important qualities of a speech synthesis system are naturalness and Intelligibility. Naturalness describes how closely the output sounds like human speech, while intelligibility is the ease with which the output is understood. The ideal speech synthesizer is both natural and intelligible. Speech synthesis systems usually try to maximize both characteristics. The two primary technologies for generating synthetic speech waveforms are concatenative synthesis and formant synthesis. Each technology has strengths and weaknesses, and the intended uses of a synthesis system will typically determine which approach is used.

1.1.5 Speech Recognition

Speech recognition is the process of converting an acoustic signal, captured by a microphone or a telephone, to a set of words for documenting or taking some action on the basis of the recognized words [10]. The recognized set of words can also serve as the input to further linguistic processing in order to achieve speech understanding. Speech recognition systems can be characterized by a number of parameters. An isolated-word speech recognition system requires that the speaker pause briefly between words, whereas a continuous speech recognition system does not. Spontaneous, or speech produced extemporaneously or speech used in normal conversations are quite different from speech read from script. The earlier contains disfluencies, and is much more difficult to recognize than the latter. Some of the speech recognition systems require speaker enrollment. In this case a speaker must provide samples of his or her speech before he or she can be correctly transcribed or interpreted. Other systems do not require any enrollment and are said to be speaker-independent. Some of the other parameters depend on the specific task. Recognition becomes more profound when the size of vocabulary is large or has numerous words with similar acoustics or sounding nature. When speech is produced in a sequence of words, language models or artificial grammars are used to restrict the search space of combination of words. The simplest language model can be specified as a finite-state network, where the permissible words following each word are mentioned explicitly. These language models generally approximate natural language in terms of a context-sensitive grammar. The difficulty of the task in this case is generally measured by perplexity based on the vocabulary size and language models. Perplexity is defined as the geometric mean of the number of words that can follow a word after the language model has been applied. There are also some external parameters that can affect speech recognition system performance. These include characteristics of the environmental noise and the type and the placement of the microphone. Due to the large number of sources of variability associated with the signal, speech recognition becomes a difficult proposition. The smallest sound units of which words are composed are called phonemes. The acoustic realizations of phonemes are greatly influenced by the context in which they appear. Furthermore, these phonetic variabilities are exemplified by the acoustic differences of the phoneme *t* in *two*, *true*and butter in American English. At word boundaries, contextual variations can be quite dramatic—making gas shortage sound like gash shortage in American English, and *devo* sound like *devandare* in Italian. Moreover, acoustic variabilities can result from changes in the environmental conditions as well as in the position and characteristics of the transducer. Other factors which can result into acoustic variabilities are the speaker's physical and emotional state, speaking rate, or voice quality. Finally, differences in sociolinguistic background, dialect, and vocal tract size and shape can contribute to speaker variabilities. As far as practical applications of speech recognition are concerned, these include applications like voice dialing (e.g., "Call home"), call routing (e.g., "I would like to make a collect call"), simple data entry (e.g., entering a credit card number), domotic appliance control and content-based spoken audio search (e.g., find a pod cast where particular words were spoken), speech-to-text processing (e.g., word processors or emails), preparation of structured documents (e.g., a radiology report), in aircraft cockpits usually termed as Direct Voice Input and even in cars to carry out small tasks like switching on the audio system [11].

1.1.5.1 Speaker Recognition

Speaker recognition is closely related to speech recognition which is mainly concerned with the problem of recognizing the speaker from a spoken utterance. Many of the techniques discussed in the context of speech recognition are also relevant in case of speaker recognition. Speaker recognition can be broadly divided into two sub tasks namely speaker verification and speaker identification. Speaker verification is concerned with the problem of verifying whether a particular speech segment under trial belongs to a specific speaker or not. It may be used to assert the claimed identity of a person or verify an accused in case of a disputed speech segment. In this thesis, we mainly discuss the latter aspect from encrypted speech point of view. On the other hand, speaker identification is mainly concerned with the problem to determine who is speaking at a particular moment from a known set of speakers. In this case, the speaker identification system makes a basic assumption that the speaker is one in a fixed group of speakers. Speaker recognition has also been discussed in this thesis from encrypted speech point of view. Several useful biometric signals exist today including retina, face, fingerprint, and voice. In the field of biometrics, voice recognition has recently been introduced as an emerging metric. There are two main inherent strengths of voice which makes it as an attractive measure for biometrics. Speech is a natural form of communication and users are not threatened by providing its samples for analysis. It may be collected without even the speaker being aware of. Secondly, due to the ubiquity of the telephone system, no specialized equipment is required to gain access to the voice.

Speaker recognition applications can be grouped into two categories namely text-dependent and text-independent speaker recognition systems. In case of text-dependent applications, the system assumes to know what text is to be spoken and it is expected that the speaker will utter the same text. On the other hand, text-independent systems offer flexibility by obviating the user to speak specific text.

An important stage in case of speaker recognition is the feature extraction stage in which speaker dependent information is extracted from the speech signal. Nearly all speaker recognition systems employ similar feature extraction mechanisms, the difference mainly lies in how these features are used. The main factor which distinguishes every speaker from other speakers is the combination of the speakers vocal tract and his/her speaking style. In almost all speaker recognition systems a training or enrolment phase is employed in which a statistical representative model of the speaker's vocal tract and speaking style is created which is generally known as the speaker model or voiceprint. Another stage is the test test phase in which the acquired speech segment is compared to one or more statistical speaker models to probabilistically determine whether it matches the models or not [12]. In this thesis, we introduce a new feature of speech when encoded with variable bit encoders i.e. packet size and use it for speaker recognition and verification in case of encrypted speech.

1.1.6 Speech Encryption

Although speech encryption is not generally considered a part of the speech processing techniques but since the main topic of this thesis is the study of encrypted speech and its properties, hence we consider speech encryption as one of the speech processing techniques. From encryption point of view, speech signal can be treated differently in the analog and digital domain. Although, in this thesis, we mainly discuss encrypted speech in the digital domain, yet a brief description of the analog speech encryption mechanisms generally known as speech scramblers is also presented.

1.1.7 Speech Scrambling

Speech scrambling deals with the analog speech signal. The analog waveform is manipulated in such way so as to make it unintelligible in a format which can be reversed by the intended recipient. The most basic form of speech scrambling is speech inversion. Speech inversion simply works by turning a signal 'inside out', i.e. reversing the signal around a pre-set frequency. There are three basic types speech inversion techniques namely, base-band or phase inversion, variable-band or rolling phase inversion and split band inversion [13].

1.1.7.1 Baseband Inversion

In this case the signal is always inverted around the same pre-set frequency. Because the frequency never changes, its makes base-band inversion highly insecure. Running the frequency through another inverter set on the same frequency unscrambles it. Descrambling baseband inversion is trivial.

1.1.7.2 Band Inversion

In this case the speech signal is inverted around a constantly varying frequency. This is also not secure but not as simple to break as baseband inversion. This type of inversion can be identified by observing the inverted signal. The burst of modem noise at the beginning of the transmission and the repeated clicking sounds as the inverting frequency changes gives a clear indication of band inversion.

1.1.7.3 Split Band Inversion

Split band inversion divides the signal into two different frequencies and inverts them separately usually through baseband inversion. Enhanced security can be attained using split band inversion systems by randomly changing the frequency where the signal is split at given intervals. It is better than baseband inversion but still far from perfect.

1.1.8 Encrypting Digitized Speech

Digitized speech signals are generally considered as ordinary binary bit streams as far as encryption is concerned. The properties of speech signals are different from text, image, video and other non speech signals and need special attention while being encrypted. These peculiar properties of speech signals have to be kept in mind not only when designing encryption algorithms, but more importantly, when these encryption algorithms are implemented in software and hardware, to secure digitized speech signals. These special properties of speech are exploited in the recently proposed encryption techniques particularly designed for speech signals. Partial or selective encryption techniques have been introduced in which the perceptually relevant portions of the speech signals are encrypted while the perceptually irrelevant portion are sent in plain in order to conserve computational resources and battery life. These encryption techniques are particularly focussed in this thesis. Two types of encryption techniques for speech encryption, like other data encryption, namely block and stream ciphers: Block ciphers tend to simultaneously encrypt groups of characters of a plaintext message using a fixed encryption transformation. Stream ciphers encrypt individual characters (usually binary digits) of a plaintext one at a time, using an encryption transformation which varies with time [14]. In the real time speech processing applications stream ciphers are preferred over block ciphers. For these applications if block ciphers are intended to be employed these are generally employed in the stream cipher modes normally called the counter modes.

Following are a few examples of the encryption systems designed for encrypting digitized speech.

1.1.8.1 Digital Voice Protection

Digital Voice Protection (DVP) is a proprietary speech encryption technique used by Motorola for their higher-end secure communications products. DVP is considered to be very secure. DVP sounds like data being passed.

1.1.8.2 Secure Telephone Unit

STU III (Secure Telephone Unit, Generation III) is the U.S. Government's standard for voice encryption. STU IIIs utilize the NSA Skipjack encryption algorithm, which is considered secure against most eavesdropping but is supposedly backdoored by the NSA. STU IIIs are considered incredibly secure, hence are restricted to government and military users, along with their civilian contractors. STU IIIs are manufactured by Lucent, L-3 Communication Systems, General Dynamics, Lockheed Martin, AT&T and Motorola. STU III compatible fax equipment is manufactured by Ilex.

1.1.8.3 STE

The NSA has finally realized that high-bandwidth telecom applications (like ISDN and DSL) are all the rage, and decided to update their crypto systems accordingly with the development of STEs (Secure Terminal Equipment). STEs provide a platform for secure 'high-speed' (128k) data allowing for video-conferencing, fax and voice applications. Unlike STU IIIs STEs have their crypto engine on a PCMCIA card, so they can be distributed more freely. STEs are manufactured exclusively by L-3 Communication Systems.

1.1.8.4 PGPfone

PGPfone is another offering from Pretty Good Privacy Inc., a secure voice program for the PC. The interface is pleasantly intuitive, and there are options for different encoders and decoders (for either cellphone or landline use). PGPfone offers a selection of encryption schemes: 128 bit CAST key (a DES-like crypto system), 168 bit Triple-DES key (estimated key strength is 112 bits) or 192 bit Blowfish key (unknown estimated key strength).

1.1.8.5 Nautilus

Nautilus is a free secure communications program. Its lacks many of the features of other communications programs, and its interface is best described as user-hateful. Unlike most other voice encryption programs, Nautilus uses a proprietary algorithm with a key negotiated by the Diffie-Hellman Key Exchange.

1.1.8.6 Speak Freely

Speak Freely is a versatile, simple voice encryption system. Speak Freely offers a selection of voice encryption techniques (IDEA or DES). Speak Freely also permits conferencing, and contains several other useful functions. Unlike most voice encryption platforms, Speak Freely includes options that it to connect to other encrypting and non-encrypting internet telephones.

1.1.8.7 AES in Counter/f8 Mode

The advanced encryption standard has also been used in the counter and f8 modes for speech encrypt particularly in Voice over IP applications. In the counter mode, which is the default encryption mode, an integer incremental counter is used with the default encryption key length of 128 bits and a default session salt key length of 112 bits.The f8 mode is a variation of the output feedback mode with a more elaborate initialization and feedback function. It uses the same default sizes for session key and salt as the AES segmented integer counter mode.

1.2 About This Thesis

In this these, we present that how the peculiar properties of speech can be utilized to extract, either directly or infer indirectly, some information from the encrypted speech signals that can be later on used for deducing important (cryptanalytic) information about the cryptographic keys, the plaintext and/or the speaker's identity. The relevant background information is presented at the start of each chapter. An important characteristic of speech signal is the presence of silence between words in a one way communication and when one speaker speaks, the other listens, in a two way conversation. We present that the linear complexity of a stream ciphered speech communication in case of silence and speech zones is different. This difference can be exploited to identify silent and speaking zones in an encrypted speech signal. It can also be utilized to extract the portions of the keystream corresponding to the silence zones. Using some of the published decimation cryptanalytic techniques it can also be used to infer the portions of the keystream corresponding to the speech zone. We also study some of the published cryptanalysis techniques particularly used for text based data and then study the effectiveness of these techniques if the underlying plaintext data is digitized and compressed speech. We also discuss a particular situation in stream ciphers called the two time pad situation in which two different plaintext are encrypted with the same keystream: a situation discussed at length in the literature with respect to text based data. We present techniques for exploitation of two time pad situation in case of stream ciphered digitized speech. Separate techniques for exploitation are presented for waveform (uncompressed) and parameter encoded (compressed) speech. In the waveform encoding domain, speech recognition approaches based on hidden Markov models can be successfully employed to decipher the two speech signals encrypted with the same keystream. In the compressed speech domain, we present that the inter frame correlation between the different components of speech frames can be exploited to reconstruct the two speech signals stream enciphered with the same keystream. The concepts of language modeling and smoothing are adapted to be deployed on the inter frame correlation between the different components in a speech frame. The mostly deployed speech compression mechanism based on code exited linear prediction, is analyzed and reconstructed from the two time pad. With the increased demand for content protection and privacy with low computational load and power consumption, particularly in mobile multimedia applications, the peculiar properties of speech are utilized to selectively encrypt some portions of the speech signal while transmitting the other portions in plain. We study the selective encryption techniques and present that these techniques may provide content protection but may not conceal the speaker's identity. A particular selective encryption technique, considered secure so far, applied over the mostly deployed speech codec, based on code exited linear prediction, is analyzed. By applying the hidden Markov modeling and Gaussian Mixture modeling techniques on the perceptually irrelevant portions of the speech signals we were able to identify the speakers with a plausible accuracy without any attempt to decrypt the communication. The recent advancement in telecommunication technologies in the form of voice over IP have brought along a lot of security risks that concern the users of the current VoIP infrastructure. Since the IP traffic has to pass through untrusted networks before it reaches its destination, it is now well understood that all VoIP traffic has to be encrypted prior to its transmission over the internet. In order to conserve bandwidth keeping in view the peculiar nature of speech conversation, the concept of variable bit rate encoding has been introduced. Although, this variable bit rate encoding, resulting into transmission of variable size packets over the internet saves considerable amount of network resources, recent studies have shown that these may also leak some important information about its contents even in case of encryption. In this thesis, we also present that these variable bit rate encoded packets can be used to identify the speaker even without decrypting the speech content. We show that the machine learning and classification techniques such as support vector machines and ensemble of nested dichotomies, which are generally popular with data mining community, can outperform the HMM and GMM approaches in case of speaker identification and verification from encrypted VoIP traffic. We also introduce the concept of classification via regression in case of speaker verification. Since, in case of speaker verification, the decision of verification is based on the threshold beyond which an unknown utterance is attributed to the hypothesized speaker we present that the regression techniques can be tricked to calculate the scores for the true speakers and imposters.

1.2.1 Thesis Layout

The general outline of the thesis and the contributions presented in different chapters are summarized below:

- In Chapter 2, we present how the statistical properties of speech can be exploited for ciphertext only cryptanalysis of stream ciphered digitized speech. Both the main speech coding techniques (waveform coding and parameter coding) have been discussed in detail. Our contribution is to show that both the speech signal properties can be used for cryptanalysis in case of stream encipherment of the speech signal even in the highly compressed environment. The main emphasis of this chapter is on the exploitation of silence zones in single channel communication and silent channels in time division multiplexed communication. Steps for resisting such exploitation are also presented in detail. Parts of the results presented here are based on [15].

- Chapter 3 presents exploitation of speech signal properties in a specific scenario pertaining to all stream ciphers i.e. the keystream reuse situation. This chapter presents details about this problem which is also known as the two time problem and has remained there for several decades, yet the threat to real systems has still been under-appreciated. The exploitation techniques presented so far in the literature in this domain deal with uncompressed text based data and by relying on the literature survey in this field one can safely say that keystream reuse in case the underlying plaintext is speech does not pose any serious threat to the security of the communication system as no such exploitation techniques are publicly known. Our contribution in this regards is to present techniques based on hidden Markov model based speech recognition for the automated cryptanalysis of two time pads in case of waveform encoded speech. The results presented here are based on our works presented in [16, 17, 18].

- Chapter 4 presents the main contribution of this thesis. It is an extension of Chapter 3 but the exploitation techniques presented over here are not based on speech recognition but on speech synthesis. It has been shown that the present day speech coding techniques are mainly based on different implementations of code exited linear prediction(CELP). Almost all modern speech processing and communication equipment including voice over IP (VoIP) employ this technique for speech encoding. In this chapter we present that how the Markov models can be used to automatically recover the two speech signals stream ciphered with the same keystream. The concept of using language models for the purpose of modeling speech parameters has been introduced for the first time in this chapter. The conventional Viterbi algorithm used for finding the internal states of an HMM from the observation symbols is modified to make it applicable in this specific scenario. The results presented over here are based on [19].

- Chapter 5 presents the applicability of the network speech recognition technique for the purpose of automated cryptanalysis of two time pads in case of speech. The three automatic speech recognition (ASR) techniques in the mobile environment namely Embedded ASR, Distributed ASR and Server based ASR are studied and their applicability for the purpose of cryptanalysis in the keystream reuse situation has been analyzed. The server based ASR has been found to be directly applicable in the situation of two time pad of stream ciphered speech signals. The two different architectures of network speech recognition namely ASR from decoded speech and ASR from Codec parameters are studied in detail and their application in the two time pad situation along with experimental results are presented. Part of the results presented in this chapter are based on [20].

- In Chapter 6, we discuss the latest selective or partial encryption techniques applied on compressed speech signals particularly in wireless multimedia applications. We present that how effective these techniques are from the perspective of confidentiality and privacy. We show that that these techniques may provide content protection but in situations where the identity of the speakers is also required to be protected then these techniques display some limitations.The results presented in this chapter are mainly based on [21].

- Chapter 7 presents flaws in the latest techniques of encryption applied over voice over IP communications. We discuss the variable bit rate encoding and its effects on the security of the speech transmitted across the internet in the form of IP packets. Apart from language and partial contents identification, we show that speaker identification and verification can be effectively done even on the encrypted packets without being deciphered. The variable length packets produced as a result of bandwidth saving variable bit rate speech encoding can be used to extract the identity of the speaker with 75 percent accuracy. The results presented in this chapters are based on [22].

- In Chapter 8 we conclude and summarize the contributions presented in this thesis with directions for future research and deliberations in the area of secure speech communications.

Chapter 2

EXPLOITING SILENCE FOR CRYPTANALYSIS

2.1 Introduction

An important characteristic of speech communication is that it is half duplex by nature, one person speaks while the other is silent. There are also silence zones between sentences and words with no speech in either direction. Measurements show that the communication circuits used for carrying speech are idle or silent 60 to 70 percent of the time, averaged over a large number of busy trunks [23]. In case of digitized speech, the presence of silence zones and the way these silence zones are encoded affect the distribution of zeros and ones in the digital bit streams. Silence between two words is generally encoded with a fixed pattern usually $(01)^*$ occasionally a 1 is replaced with a 0 and a 0 replaced with a 1 [24]. In case of multiple channel communications, a channel itself may be silent or idle completely. The silent part of digitized speech is non random in nature and portions of the digitized speech corresponding to words are random in nature. These random and non random strings of bits, when encrypted using a linear feedback shift register based stream cipher, display different statistical properties. Amongst different statistical tests for distinguishing between cipher text corresponding to the random and non random bit streams, the linear complexity profile test has shown significant difference between the two [24] and can be used effectively for this bifurcation. These silence zones are randomly placed in the overall bit stream and in order to mark the silent and non silent zones in the cipher text, the techniques presented in [4] can be employed. The different types of ciphertext only attacks on stream ciphers

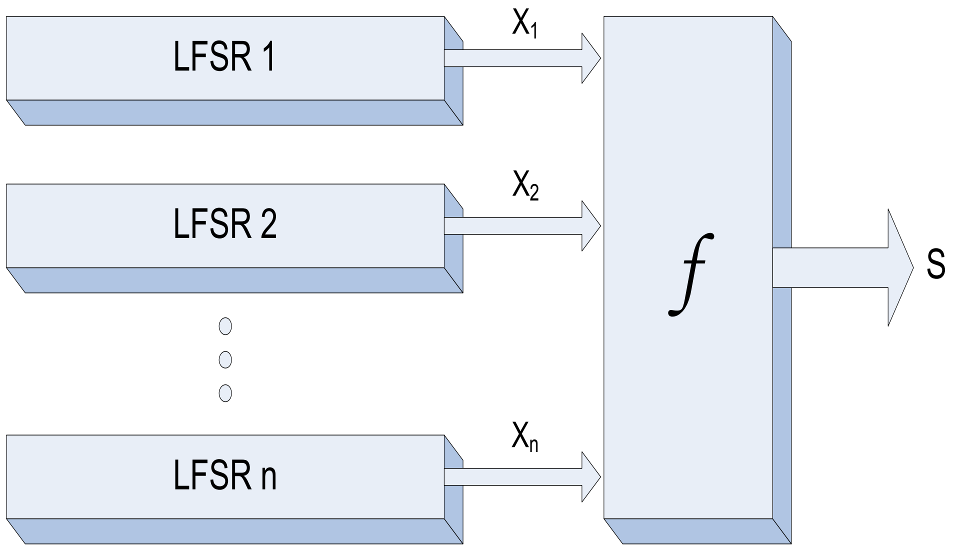

Figure 2.1: LFSR based stream cipher using Combination Generator

presented uptil now e.g. [25, 26, 27] assume the plain text to be statistically biased i.e. $P_0 = 0.5 + \epsilon$. and all these attacks fail if the plain text is not statistically biased i.e. $P_0 = P_1 = 0.5$. We present techniques for the cipher text only attacks on LFSR based stream ciphers even if the plaintext is not statistically biased but assuming it to be speech. A generalized form of stream cipher consisting of several LFSRs combined by a non linear Boolean function as shown in figure 2.1 is considered.

2.2 Related Work

No previous published work on exploitation of silence zones for the purpose of cryptanalysis exist, however, previously distinguishing attacks have been discussed but in a different context [28, 29]. In these types of distinguishing attacks no distinction is carried out between the portions of ciphertext corresponding to a random plain text and portions of the cipher text corresponding to a non random plain text. In all these attacks the distinction is made between the output of a specific cipher and that of pure random bit stream selected from a uniform distribution. Moreover, no significant cryptanalytic information is deduced during this distinction [30]. Large linear complexity has been a necessary but not sufficient condition for randomness of binary streams since long [31]. The use of linear complexity profile for distinguishing between random input ciphertext and non random input ciphertext has been introduced for the first time in an unpublished work by P. Chandra Sekhar in his masters thesis [24]. The linear complexity profile test for breaking ciphertext into portions corresponding to silence zone and speaking zones has been used by Sumankumar Pramanik, again in unpublished work of his MS thesis [4]. Although the linear complexity profile tests have been conducted and experimental details are presented but the mathematical reasoning for this has not been discussed.

2.3 Linear Complexity as Silence Distiguisher

The linear complexity of a binary stream is the minimum length of the LFSR which can generate it. Large linear complexity has been an established necessary but not sufficient condition for randomness of binary streams. A non random binary stream, like in case of silence zones encoding stream (01)* have very low linear complexities, say 2 in this case. In case of LFSR based stream ciphers, the keystream, which is a pseudorandom bit stream, has generally large but finite linear complexity which is added modulo to the plain text streams. The addition of two binary streams with different linear complexities is governed by the following proposition.

2.3.1 Proposition

Let P and Q be two non constant polynomials over F2 and let S(P) denotes the set of all sequences produced by the LFSR with feedback polynomial P then we have

$$(ut + vt)t > 0, u \in S(P), v \in S(Q) = S(R)$$

where R is the least common multiple of P and Q [25, 30].

This proposition leads to the following corollary.

2.3.1.1 Corollary

Let $(u_t)t > 0$ and $(v_t)t > 0$ be two binary sequences with linear Complexity L_1 and L_2 respectively, then the binary sequence produced by the addition modulo 2 of u and v i.e. $(u_t + v_t)t > 0$, has a linear complexity L where $L = L_1 + L_2$.

Now, the silence zone encoding, being a fixed pattern repeated at regular and short intervals, has very low linear complexity as compared to the speech zone. This low linear complexity sequence when added to the keystream with finite linear complexity will result in another low linear complexity sequence. In case of speech zone, the linear complexity is generally very large. This sequence when added to a finite and large linear complexity sequence results in a very large linear complexity sequence. Hence linear complexity analysis can be used as an effective tool for distinguishing between cipher text corresponding to silence zone and cipher text corresponding to speech zones in stream ciphered digitized voice.

2.4 Exploiting Silence in Single Channel Communication

In this section, we present how to convert a ciphertext only attack to a known plain text attack in case of single channel communication. Both possibilities of speech coding i.e. waveform(uncompressed) as well as predictive(compressed) coding are discussed.

2.4.1 Waveform (uncompressed) Coding

In this case, contrary to the conventional ciphertext only cryptanalysis of LFSR based stream ciphers, the plain text need not be statistically biased. The silence zones are encoded as fixed patterns of 1s and 0s. Table 2.4.1 shows the encoding of the silence zones according to different

Table 2.1: Silence Zone Encoding in Various Speech Coding Standards

S No.	Encoding Standard	Silence Zones Coding
1	PCM 8kHz 8 bit	0x 80 and 0x 7F
2	CCIT μ-Law	0x FF
3	CCIT A- Law	0x 55 and 0x D5
4	ITU-T G.722	0x 7F and 0x FD

standard encoding techniques. By the linear complexity profile analysis presented in [4] we can mark the silence and speech zones in the cipher text. In a typical speech file the length of the silence zone is approximately 3000 samples [4] which correspond to 24000 bits in case of 8 bits/sample encoding and 48000 bits in case of 16 bits/sample encoding. Finding the silence zone in the ciphertext is equivalent to finding the keystream, thereby converting a ciphertext only attack to a know plain text attack. In [4] a complex mechanism for breaking the cipher without even finding the keystream has been presented. A simple and more straight forward approach would be to use the well established and efficient techniques of fast correlation attacks [24, 4, 25] to find the internal states of the LFSRs. The keystream can then be calculated from that point onwards and by just XORing the keystream and the cipher text the plain text can be deduced.

2.4.1.1 Experimental Results

We consider the following example of combination generator consisting of three LFSRs combined by the non linear Boolean function

$$f(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3$$

and having connection polynomials as

$$P_1(x) = 1 + x^2 + x^4 + x^5 + x^6 + x^8 + x^9 + x^{10} + x^{11} + x^{13} + x^{15}$$

$$P_2(x) = 1 + x^2 + x^4 + x^5 + x^6 + x^8 + x^9 + x^{10} + x^{11} + x^{13} + x^{14} + x^{15} + x^{17}$$

and

$$P_3(x) = 1 + x + x^7 + x^8 + x^{10} + x^{12} + x^{15} + x^{16} + x^{17} + x^{20} + x^{21} + x^{22} + x^{23}$$

The linear complexity of the keystream produced by this stream cipher is 991. A typical speech file encoded with PCM (8000 samples per second 8 bit per sample) is encrypted with it. The linear complexity of the silent zones comes out to be 993 whereas those of the speech zones are greater than 5000 in all cases. Hence linear complexity can be used effectively to identify the silence and speech zones in an encrypted digitized speech file. About 25600 contiguous bits of the keystream can be deduced from the silence zones, thereby converting a ciphertext only attack to a known plain text attack without using the statistical bias of the plain text. Applying the fast correlation attacks [32, 33], the initial states of the LFSRs can be deduced and the keystream can be calculated from that point onwards. XORing the keystream and the ciphertext, the plain text can be found out.

2.4.2 Coding Schemes with Compression

Silence in speech communication can also be exploited in case when compression is applied on the speech while digitizing and encoding it. In this case, speech is linearly modeled and instead of transmitting the quantized waveform, the quantized parameters of the linear model are transmitted along with residual error between the actual and modeled speech [2]. In this case we deal with frames which are about 20-30 millisecond duration of speech. The frames corresponding to silence zones when encoded through linear predictive coding schemes also are fixed and have a particular pattern which is repeated in each of the silence zone in the speech signal. The concept of linear complexity profile analysis again become applicable.

2.4.2.1 Experimental Results

Speech is encoded using a CELP encoder at 9.5 kbps. In this case, a frame consisting 160 samples of speech is encoded as 190 bits. A frame corresponding to silence zone is always encoded with a fixed pattern of bits comprising of 107 zeros and 83 ones. The linear complexity of one frame is 95 while that of more than one frame is 189. A typical silence zone comprises of about 18 frames which corresponds to 3420 bits of silence zone having linear complexity as 189 stabled after 377 bits. The same speech is enciphered with a stream cipher of example 1. The linear complexity of the cipher text corresponding to silence zone is 1180 whereas that of the speech zone is greater than 5000.

2.5 Exploiting Silence in Multiplexed Communication

In this section, we present a technique to exploit silent channels in case of time division multiplexed communication systems. Time division multiplexed systems are widely used in telecommunication. The E1/T1 lines being the basic standards being used by the US and European community for Time division multiplexing of voice and data channels. The TDM scheme of multiple channels transmission has also being adopted by the fiber optic community and the conventional E1/T1 links have been replaced by the SONET and SDH [34]. Here we will confine to the voice channels only. In the multiplexed communication environment the silence zones may also exist at the individual channel level and the techniques of cryptanalysis discussed in section 2.4 would still be applicable on these provided single channel encryption is carried out. However, if compression is applied at the individual channel then our scheme of exploiting silence zones will not work in this case directly; however since all the channels are not always active, we can in this case exploit the presence of silent/idle channels. In case the channel is silent then a fixed pattern of zeros and ones is being transmitted in the TDM slot of the silent channel. The framing information is not encrypted generally. The reason for this is the synchronization of the multiplexer and the de-multiplexer. From the framing information, the channels can be demultiplexed and then the silent channels can be identified using the techniques of linear complexity analysis of [4]. We assume that the fixed pattern being transmitted in case of silent channels is standard and generally known. By XORing that fixed pattern with the cipher text the keystream portion corresponding to the silent channels can be found out. It is worth mentioning here that this key stream does not correspond to a contiguous portion of the key stream but it is a decimated form of the key stream. The factor of decimation depends on the position of the silent slot in the TDM frame. By applying the decimation attack on the stream cipher technique presented in [27], the LFSR state can be deduced using the decimated keystream. In this case the position of the silent channel in the TDM frame determines the decimation factor d.

2.5.1 Experimental Results

Consider an eight channel multiplexed communication link as shown in figure2.2. Channel 2 is the silent channel i.e. no voice communication is being transmitted on channel 2, instead a non random pattern (say $(01)^*$) is transmitted on this channel. The four frames shown in the figure are encrypted using a stream cipher of Example 1 as a bulk encryptor before the addition of the framing information. The plain text stream in this case is represented as $C_k[t]t > 0$, where k is the channel number. The plain text binary stream is

$$C_1[1], C_2[1], C_3[1], C_4[1], C_1[2], C_2[2], \ldots$$

The keystream is represented as

$$K[1], K[2], K[3], K[4], K[5], \ldots$$

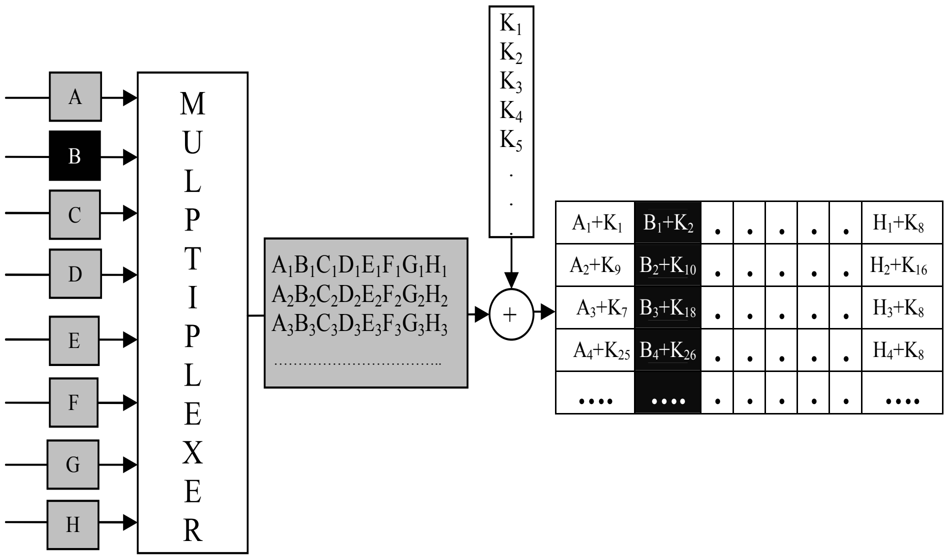

Figure 2.2: Time Division Multiplexed Communication with Channel 2 being Silent

The ciphertext becomes

$$C_1[1] \oplus K[1], C_2[1] \oplus K[2], C_3[1] \oplus K[3], C_4[1] \oplus K[4], C_1[2] \oplus K[5], C_2[2] \oplus K[6], \ldots$$

Since the framing information is not encrypted hence the encrypted frame can be demultiplexed. Applying linear complexity profile analysis of [4] on the four channels separately the silent channel 2 shows a significantly low linear complexity (i.e. 992) as compared to channel 1,3 and 4 (linear complexity ¿ 5000) and can easily be identified as silent. Since we know the non random pattern transmitted by the silent channel, thereby we can find out the keystream bits with respect to this channel (by XORing the cipher bits and the non random pattern). The keystream bits K[2], K[6], K[10],correspond to a regularly decimated form of the keystream K with decimation factor 4. Applying the decimation attack of [27] with a slightly different concept, the initial states of the LFSRs can be deduced.

2.6 Resistance to Silent Zones Exploitation

The biggest assumption in case of exploitation of silence zones for cryptanalysis is that the linear complexity of the stream cipher used is finite. This assumption is generally valid as there is a requirement for a large linear complexity on one side but on the other hand a large correlation immunity is also mandatory. Since there exists a trade off between linear complexity and correlation immunity [35], hence the linear complexity cannot be increased beyond limits. The other assumption deals with the encoding scheme and the size of the silence zones in terms of number of bits, which is related to the sampling rate while digitizing speech and the number of bits per sample as well. The larger the resolution and the sampling rate, the larger would be the size of the silence zone and the more will be the cipher susceptible to such attack. A criterion for resisting such types of attacks would be that the size of the silence zones should be less than twice the linear complexity of the stream cipher in case of single channel communication. In case of multiplexed communication, it is not the silence zone which is exploited but the silent/idle channels are used for this purpose. One way to resist the attack would be to send a random pattern of data on the silent channel instead of transmitting a regular pattern which makes the distinguishing attack on the basis of linear complexity profile more difficult. Another way to resist this type of attack would be to use Digital Signal Interpolation [23]. Although, the purpose of digital signal interpolation is efficient utilization of bandwidth but since it inserts data of other channels into the silence zones thereby removing the silence zones and resisting this attack. Statistical time division multiplexing will also resist such attacks.

2.7 Summary

In this paper we have presented analysis of silence zones, for the purpose of cryptanalysis, the speech zones also have specific properties which can be exploited for this purpose. The preliminary work in this regards has been presented in [4]. The algorithm presented here can be further improved by the addition of dynamic correction capability. Techniques other than linear complexity for the purpose of classifying the silent and speaking zones in the cipher text also needs to be looked into. In case of very large linear complexities the, i.e. if the linear complexity is larger than half the length of the silent zone then the linear complexity analysis test will fail. The asymptotic analysis of the classification of silence zones and noise zones as well as that of the cryptanalysis on the basis of this technique also needs to be investigated. Comparisons with the other published cryptanalytic algorithms from the point of view of resources and time also require some attention. The statistical properties of digitized speech are quite different from other digital data such as text, image and video. The half duplex nature of speech can be exploited for the purpose of cryptanalysis as well. Linear complexity profile can effectively be used to identify the silent zones in single channel silent/idle channels in multiple channel communication systems. By using the silent channel encoding, a cipher text only attack can be effectively converted to a known plain text attack, even if the plain text is not assumed to be statistically biased. These limitations of digitized speech are to be kept in mind while applying encryption on communication links carrying speech data, in order to avoid exploitation of silence zones and silent/idle channels.

Chapter 3

CRYPTANALYSIS OF TWO TIME PADS OF WAVEFORM ENCODED SPEECH

3.1 Introduction

In a stream cipher, a message m is exclusive ORed with a keystream k to produce the ciphertext c i.e. $c = m \oplus k$. If the keystream k is random and is of the same size as that of the message m then the stream cipher becomes a one time pad which is considered as a perfect cipher [36] in the cryptographic community. If two different plaintexts m_1 and m_2 are encrypted with the same keystream k then their results $m_1 \oplus k$ and $m_2 \oplus k$ can be XORed to neutralize the effect of the keystream k, thereby obtaining $m_1 \oplus m_2$. The key reuse problem in stream ciphers and its exploitation in different scenarios for uncompressed text based data have been studied since long. It has also been mentioned in the literature as the two time pad problem [37]. The vulnerability of keystream reuse exists with many practical systems which are still in use such as Microsoft Office [37, 38], 802.11 Wired Equivalent Privacy (WEP) [39], WinZip [40], Point to point tunneling protocol (PPTP) [41] etc. In addition to this, the two time pad problem is predicted to remain there for quite some time in the near future also because of the endorsement of counter mode of AES by NIST [37, 42] for high speed data transfer applications. In this case, cryptographers, who would have otherwise used a block cipher with cipher block chaining (CBC) mode, are compelled to use AES in the counter mode, thereby turning a block cipher into a stream cipher and the chances of reusing key streams is further enhanced. Also, 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 [43, 44]. This has further increased the possibility of keystream reuse in actual systems because an effective and secure key management has still been an uphill task for the crypto designers as mentioned in [45] as: *If you think you know how to do key management, but you don't have much confidence in your ability to design good ciphers, a one-time pad might make sense. We're in precisely the opposite situation, however: we have a hard time getting the key management right.. And almost any system that uses a one-time pad is insecure. It will claim to use a one-time pad, but actually uses a two-time pad (oops).* The keystream reuse exploitation techniques discussed so far in the literature assume the underlying plaitxt to be uncompresed text-based data source encoded hrough conventional encoding mechanisms such as ASCII coding etc. All the exploiattion techniques which are mainly based on heuristic rules [46, 47] and even those based on mathematical modeling [37] fail if the the underlying data is not uncompressed text. Keeping in view the literature presented so far on this topic one can safely reuse key streams if one wants to stream encipher speech based data. In this chapter we present the first ever use of hidden Markov model based speech recognition approach to automated cryptanalysis of two time pads of speech based data. Speech is digitized, encrypted and sent between two parties in many situations. Encryption schemes particularly designed for speech whether these are analog speech scramblers (e.g. [48]) or modern digital selective speech encryption techniques (e.g. [49]), have 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 [50] and automatic transcription of conversational speech [51] 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 cryptanalysis through speech recognition.

3.2 Background Information

3.2.1 Speech Coding and Compression

Speech coding is quite different from other forms of audio coding. The details of speech coding are presented in Section 1.1.3. Speech signals are composed of a sequence of sounds. These sounds and the transitions between them serve as a symbolic representation of information. In order to encode this information mainly two types of encoding mechanisms are adopted, namely waveform and parameter encoding techniques. Waveform coders attempt to code the exact shape of the speech waveform without considering the nature of human speech production and speech perception. In waveform encoding, the speech signal being a continuous-time signal is sampled, quantized (and compressed) and then digitally encoded. Speech signal is typically quantized either 16-bit uniform or 8-bit companded quatization. The A-law and μ-law algorithms [3] used in traditional Pulse Coded Modulation (PCM) digital telephony can be seen as very early precursors of speech digitization based on waveform encoding. Like many other signals the sampled speech signal contains a great deal of information which is either redundant (non zero mutual information between successive samples in the signal) or perceptually irrelevant (information that is not perceived by human listeners). In case of parameter encoding, the speech coder converst a digitized speech signal into a coded representation which is transmitted in the form of frames. A sppech decoders receives these coded frames and synthesizes them to reconstruct speech. Standards in parametr encoded speech simply dictate the input-output relationships of both coder and decoder. In case of parameter encoding speech is considered as a source filter model in which the parameters of the model alongwith excitation information in the form of voiced/unvoiced signals is used for digital representation of speech. The transfer function of that model is assumed to be all-pole (autoregressive model). The excitation function is a quasiperiodic signal constructed from discrete pulses (18 per pitch period), pseudorandom noise, or some combination of the two. If the excitation is generated only at the receiver, based on a transmitted pitch period and voicing information, then the system is designated as an LPC vocoder. LPC vocoders that provide extra information about the spectral shape of the excitation have been adopted as coder standards between 2.0 and 4.8 kbps. LPC-based analysis-by-synthesis coders (LPC-AS), on the other hand, choose an excitation function by explicitly testing a large set of candidate excitations and choosing the best. LPC-AS coders are used in most standards between 4.8 and 16 kbps. Subband coders are frequency-domain coders that attempt to parameterize the speech signal in terms of spectral properties in different frequency bands. These coders are less widely used than LPC-based coders but have the advantage of being scalable and do not model the incoming signal as speech. Subband coders are widely used for high-quality audio coding. In the analysis by synthesis class of speech coding, Code Excited Linear Prediction (CELP)is the most popular and is in use in most of the speech processing applications as well as digital communications of speech [52]. Table 3.1 presents some of the major standard speech coding techniques alongwith applications and data rates [53]. Although speech coders based on CELP are well known and common coders used in voice over IP networks and predominantly used in PC based systems yet some of the leading IP phone vendors unfortunately stopped supporting some implementations of CELP. This leads to

Table 3.1: Speech Coding Standards and Applications

Application	Rate(kbps)	Standard	Coding Technique
Landline Telephone	64	G.711	PCM
	16-40	G.726	ADPCM
	16-40	G.727	ADPCM
Tele conferencing	48-64	G.722	Split Band ADPCM
	16	G.728	LD-CELP
Cellular Telephones	13	Full-Rate GSM	RPE-LTP
	12.2	EFR	ACELP
	7.9	IS-54	VSELP
	6.5	Half-Rate GSM	VSELP
	8.0	G.729	ACELP
	4.75-12.2	AMR	ACELP
	1-8	IS-96	QCELP
Multimedia	5.3-6.3	G.723.1	MPLPC, CELP
	2.0-18.2	MPEG-4	HVXC, CELP
Satellite Telephone	4.15	M	IMBE
	3.6	Mini-M	AMBE
Secure Communications	2.4	FS1015	LPC-10e
	2.4	MELP	MELP
	4.8	FS1016	CELP
	16-32	CVSD	CVSD

G.711, which is based on waveform coding, as the common coder for PC to IP phones [54]. Moreover, most of the telecommunication links still use the A-law and -law algorithms of waveform coding for speech digitization. Keeping this fact in mind, we have developed our algorithms for the keystream reuse exploitation of speech signals based on both waveform as well as parameter encoding. In this chapter we will discuss the waveform encoded speech.

3.2.2 Language Modeling

3.2.2.1 Definition

A language model is a statistical model which assigns a probability to each word/alphabet of the language through a probability distribution. Language models are used in many natural language processing applications such as speech recognition, machine translation of speech, parts of speech tagging, information retrieval, optical character recognition, handwriting recognition, spell checking etc [55]. It is usually formulated as a probability distribution $p(s)$ over string s which determines how frequently a string s occurs as a sentence. For example, for language model describing spoken language we might have the probability of the string *good morning* as $p(howareyou) = 0.05$. This means that one out of every two hundred sentences a person speaks is *how are you*. An interesting fact about language modeling is that the string might be grammatically correct but its probability of occurrence might still be nearly zero. Like the probability of the string *good night* is quite high whereas that of *good mid night* might be zero, whereas both strings are grammatically correct.

3.2.2.2 n-gram Models

The most widely used language models by far are n-gram language models. These models are introduced by considering the case for $n = 2$; these models are known as *bi-gram* models. For a sentence (word) s composed of words (alphabets)$w_1, w_2, \ldots w_l$, we can write

$$p(s) = p(w_1)p(w_2|w_1)p(w_3|w_1w_2)\cdots p(w_l|w_1\cdots w_{l-1}) = \prod_{i=1}^{l} p(w_i|w_1\cdots w_{i-1}) \tag{3.1}$$

In the bi-gram models we assume that the probability of a word letter depends only the immediately preceding word letter only. In our implementation of language models we will also restrict to bi-gram models only. 3.1 for the bi-gram models can be expressed as

$$p(s) = \prod_{i=1}^{l} p(w_i|w_1\cdots w_{i-1}) \approx \prod_{i=1}^{l} p(w_i \mid w_{i-1}) \tag{3.2}$$

To make $p(w_i \mid w_{i-1})$ meaningful for i=1, we can pad the beginning of a word or sentence with a distinguished token $< BOS >$ that is we pretend $w_0 =< BOS >$. For a detailed description of language models the reader is referred to [55].

3.2.2.3 Smoothing

Language models developed from even very large corpora of text based data do not incorporate all possibilities as certain sequence may not be observed during the training process. In order to solve this inherent problem with language models, they are often approximated using smoothed n-gram models. Smoothing is a technique used to estimate better probabilities when data available is not sufficient to estimate accurate probabilities [55]. The name smoothing comes from the fact these techniques tend to make distributions more uniform by adjusting low probabilities such as zero probabilities upwards and high probabilities downwards. Not only do smoothing methods prevent zero probabilities but they also attempt to improve the accuracy of a model as a whole. Whenever a probability is estimated from few counts, smoothing has the potential to significantly improve estimation. This point has been proven in our case of implementation of smoothing techniques while modeling interframe speech parameters. Out of the many smoothing techniques available we used additive and Witten Bell smoothing techniques during our experiments which have significantly improved our results.

3.2.2.4 Additive Smoothing

It is the simplest of all the smoothing techniques. In this case we pretend that number of occurrence of a word/letter in a training data is δ times more than the actual occurrences where typically $0 < \delta \leq 1$ i.e.

$$p_{add}\left(w_i \mid w_{i-n+1}^{i-1}\right) = \frac{\delta + c\left(w_{i-n+1}^{i-1}\right)}{\delta \mid V \mid + \sum_{w_i} c\left(w_{i-n+1}^{i-1}\right)} \tag{3.3}$$

3.2.2.5 Jelinek-Mercer Smoothing

The JM smoothing is based on linear interpolation. A unigram model or 1-gram model conditions the probability of occurence of a word/letter on no other word/letter and just reflects the frequency of word/letter in a text. For example the maximum likeleihood unigram model is

$$p_{ML}\left(w_i\right) = \frac{c\left(w_i\right)}{\sum_{w_i} c\left(w_i\right)} \tag{3.4}$$

where c represents the count. We can linearly interpolate a bigram model and unigram model as follows:

$$p_{interp}\left(w_i \mid w_{i-1}\right) = \lambda p_{ML}\left(w_i \mid w_{i-1}\right) + \left(1 - \lambda\right) p_{ML}\left(w_i\right), 0 \leq \lambda \leq 1. \tag{3.5}$$

It is always useful to interpolate higher order n-gram models using lower n-gram models because when there is indufficient data to estimate a probability in the higher order model, the low order model can often provide useful information. An elegant way of performing this interpolation is as given by Brown et al. [56]

$$p_{interp}\left(w_i \mid w_{i-n+1}^{i-1}\right) = \lambda_{w_{i-n+1}^{i-1}} p_{ML}\left(w_i \mid w_{i-n+1}^{i-1}\right) + \left(1 - \lambda_{w_{i-n+1}^{i-1}}\right) p_{interp}\left(w_i \mid w_{i-n+2}^{i-1}\right) \tag{3.6}$$

The nth-order smoothed model is defined recursively as a linear interpolation between the nth-order maximum likelihood model and the (n-1)th-order smoothed model. To end the recursion we can take the smoothed 1st-order model to be the maximum likelihood distribution or we can take the smoothed 0th-order model to be the uniform distribution

$$p_{unif}\left(w_i\right) = \frac{1}{\mid V \mid} \tag{3.7}$$

where V represents the number of alphabets/words in a language. We take alpphabetsif we are modellinf letters and take words if our n-gram models represent words in a language.

3.2.2.6 Witten Bell Smoothing

Witten Bell Smoothing was developed for text compression and can be considered as a particular case of JM smoothing. In this case the nth-order smoothed model is defined recursively as a linear interpolation between the nth order maximum likelyhood model and the (n-1)th-order smoothed model as shown in 3.6

$$p_{WB}\left(w_i \mid w_{i-n+1}^{i-1}\right) = \lambda_{w_{i-n+1}^{i-1}} p_{ML}\left(w_i \mid w_{i-n+1}^{i-1}\right) + \left(1 - \lambda_{w_{i-n+1}^{i-1}}\right) p_{WB}\left(w_i \mid w_{i-n+2}^{i-1}\right) \tag{3.8}$$

To compute the parameters $\lambda_{w_{i-n+1}^{i-1}}$ for Witten Bell Smoothing, we will need to use the number of unique words that follow the history w_{i-n+1}^{i-1}. We will write this value as $N_{1+(w_{i-n+1}^{i-1}\bullet)}$, formally defined as

$$N_{1+(w_{i-n+1}^{i-1}\bullet)} = \{w_i : c(w_{i-n+1}^{i-1} w_i)\ 0\} \tag{3.9}$$

The notation N_{1+} is meant to evoke the number of words that have one or more counts and the $\bullet$ is meant to evoke a free variable that is summed over. We can then assign the parameters $\lambda_{w_{i-n+1}^{i-1}}$ for Witten Bell Smoothing such that

$$1 - \lambda_{w_{i-n+1}^{i-1}} = \frac{N_{1+(w_{i-n+1}^{i-1}\bullet)}}{N_{1+(w_{i-n+1}^{i-1}\bullet)} + \sum_{w_i} c(w_{i-n+1}^{i})} \tag{3.10}$$

In Witten Bell smoothing we use the higher order models if the corresponding n-grams occur in the training data and back off to the lower order models otherwise.Then we should take the term $1 - \lambda_{w_{i-n+1}^{i-1}}$ to be the probability that a word/letter not observed after the history w_{i-n+1}^{i-1} in the training data occurs after that history. To estimate the frequency of these novel words, imagine traversing the training data in order and counting how many times he word following the history w_{i-n+1}^{i-1} differs from the words in all such previous events. The number of such events is simply $N_{1+(w_{i-n+1}^{i-1}\bullet)}$, the number of unique words that follow the history w_{i-n+1}^{i-1}. Eq 3.10 can be viewed as an approximation of this intuition. The use of Witten Bell Smoothing in our experiments has significantly improved the accuracy of decoding speech parameters in the two time pad situation.

3.2.3 Hidden Markov Models

A Markov process is a stochastic process in which the future state of the process depends only on the present states and not on any past state. A Markov model is a model in which the system to be modeled is assumed to be a Markov Process. A hidden Markov model (HMM) is a statistical model in which the system to be modeled is assumed to be a Makov process with hidden parameters and the challenge is to find the hidden parameters of the model from the observable parameters. An HMM is characterized by the following [57]:

1. N, the number of states $S_1, S_2, \ldots S_N$ in the model. Although the states are hidden in an HMM yet in every practical system some physical significance is attached to the states. Generally the states are connected in such a way that any state can be reached from any other state. We denote individual state at at time t as q_t.

2. M, the number of distinct observation symbols per state, generally known as the alphabet size. The observation symbols correspond to the physical output of the system being

modelled. We denote the individual symbols as $V = \{v_1, v_2, \ldots, v_M\}$.

3. The state transition probability distribution $A = \{a_{ij}\}$ where

$$a_{ij} = P\left[q_{t+1} = S_j \mid q_t = S_i\right],\ 1 \leq i, j \leq N.$$

   For the special case that any state can be reached from any other state a_{ij} 0 for all i, j.

4. The observation symbol probability distribution in state j, $B = \{b_j(k)\}$, where

$$b_j(k) = P\left[v_k\, at\, t \mid q_t = S_j\right],\ 1 \leq j \leq N,\ 1 \leq k \leq M$$

5. The initial distribution $\pi = \{\pi_i\}$ where

$$\pi_i = P\left[q_1 = S_i\right],\ 1 \leq i \leq N$$

Given appropriate values of N, M, A, B and π, the HMM can be used as a generator to produce an observation sequence

$$O = O_1\, O_2\, \ldots\, O_T$$

(where each observation O_t is one of the symblos from V and T is the number of observations in the sequence) as follows:

1. Choose an initial state $q_1 = S_i$ according to the initial state distribution π.

2. Set $t = 1$.

3. Choose $O_t = v_k$ according to the symbol probability distribution in state S_i, i.e. $b_i(k)$..

4. Transit to a new state $q_{t+1} = S_j$ according to the state transition probability distribution for state S_i i.e. a_{ij}.

5. Set $t = t + 1$, if $t < T$ go to step 3, else terminate the procedure.

It is evident from the above discussion that an HMM can be completely described by two model parameters M and N and three probability measures A, B and π. For convenience we use the compact notation

$$\lambda = (A,\, B,\, \pi)$$

to indicate the complete parameter set of the model.

3.2.4 Automatic Speech Recognition

The purpose of speech recognition is to convert spoken words to machine readable input. Two main techniques of speech recognition presently exist, one based on dynamic time warping (DTW) and the other based on hidden Markov models (HMMs)[58]. Dynamic time warping is used for measuring similarity between two sequences which may differ in time or speed [59]. DTW has been used for speech recognition for a long time but has now been replaced by HMM-based approach which has proved to be more successful and this was the reason for our selection of this approach. For complete details of the hidden Markov models and their applications in speech recognition the reader is referred to [57]. All modern speech recognition tools use this technique because of its robustness, flexibility and efficiency. The goal of any ASR system is to find the most probable sequence of words/phonemes $W = (w_1, w_2, w_3, \ldots)$ given an acoustic observation $O = O_1\, O_2\, \ldots\, O_T$. Mathematically,

$$W = argmax_{i \in V} P(w_i | O) \tag{3.11}$$

where V indicates the phonetic units (generally phonemes) in a language and are modelled as HMMs. Equation 3.11 cannot be solved directly but using Bayes Rule, the above equation can be modified as

$$W = argmax_{i \in V} \frac{P\,(O \mid w_i)\, P\,(w_i)}{P\,(O)} \tag{3.12}$$

or it can also be written as

$$W = argmax_{i \in V} P\left(O \mid w_i\right) P\left(w_i\right) \tag{3.13}$$

here, $P(O \mid w_i)$ is calculated using HMM based acoustic models, whereas $P(w_i)$ is determined from the language model.

3.3 Related Work

3.3.1 Keystream Reuse Exploitation

Key stream reuse vulnerability exploitation of stream ciphers dates back to World War II. Following are some of the significant and worth mentioning works presented in the literature.

3.3.1.1 Venona Project

The US Army Signals Intelligence Unit which later on became the National Security Agency noticed that some of the soviet military and diplomatic telegraphic traffic used keystreams for encryting more than one messages. Most of the messages which would later prove to be decipherable were intercepted between 1942 and 1945, and they were decrypted beginning in 1946 and continuing until 1980, when Venona was cancelled. More than 3000 messages were at least partially decrypted [60, 61]. The Soviet systems in general used a code to convert words and letters into numbers, to which additive keys (from one-time pads) were added, encrypting the content. When used correctly, one-time pad encryption is provably unbreakable. Cryptanalysis by American and British code-breakers revealed that some of the one-time pad material had incorrectly been reused by the Soviets (specifically, entire pages, although not complete books), which allowed decryption (sometimes only partial) of a small part of the traffic. Generating the one-time pads was a slow and labor-intensive process, and the outbreak of war with Germany in

June 1941 caused a sudden increase in the need for coded messages. It is probable that the Soviet code generators started duplicating cipher pages in order to keep up with demand.

3.3.1.2 Formalization of Keystream Reuse Exploitation

Another worth mentioning work on the topic is that of Rubin in 1978, who for the first time formalized the process of keystream reuse exploitation [46]. In this work the “classical” approach of extracting p and q from p xor q is formalized assuming p and q to be taken from English text. Initially, a word is guessed that is likely to appear in the messages, for example the word commonly used word ”the”. Then, we try to xor the with each substring p and q of length equal to three alphabets. Wherever it results into something that “looks like” English text, there are fair chances that one of the messages has the in that position and the other message has the result of the xor. The same process when repeated a number of times may help in extracting sufficient words from the two plaintexts which are xored. It was Rubin if 1978 who formalized this process for the first time.

3.3.1.3 Automating the Process of Keystream Reuse Exploitation

Dawson and Neilson in 1996 for the first time automated the process of cryptanalysis of plaintext XORs [47]. They created a program which was based on a number of heuristic rules in order to automatically deduce p and q from their xor. Their approach was simplified by the assumption that p and q only belonged to the the 26 English letters that too only uppercase with the addition of the space. Since in ASCII, an uppercase letter xored with a space can not be equal to any two uppercase letters xored together, hence things get simplified further. It was further assumed that if p xor q was equal to 0, both the characters were spaces. Between the recovered spaces a list of common words of different lengths was used in conjunction with some ”tricks” to extract p and

q. They tested their system on the first 0.6 million characters of the English Bible. It is worth mentioning here that the rules adopted in this approach were designed to work well in the specific example cases and there was no guarantee they would work on different examples with the same accuracy.

3.3.1.4 Using Mathematical Models for Keystream Reuse Exploitation

It was only very recently in late 2006 that mathematical models were used for the first time for this important area of cryptanalysis of two time pads by Joshua Mason et. al. [37]. Mostly the keystream reuse exploitation discussed previously is with respect to the textual data and mainly based on heuristic rules for obtaining the two plaintexts m1 and m2 from except for [37] which uses statistical finite states language models and natural language approach. Prior works also exist on automated cryptanalysis of analog speech scramblers (e.g. [62]), 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 speech signals. In [37], the concepts borrowed from the natural language and speech processing communities are used for text based data whereas we use similar concepts with addition of speech recognition for speech based data.

3.3.1.5 Cryptanalysis of Encrypted Speech

The statistical properties of speech signals are different from text based data. It was this realization of different statitical properties that the present day selective encrytion techniques were developed which selectively encrypt portions of the speech signal while leave certain portions unencrypted and still achieve a high degree of unitelligibility. Speech encryption devides can be didde into two main categories i.e. analog encrytion (scrambling) and digitla ncrytion devices.

The analog speech scramblers can be considered as early precursors of the speech encryption domain which were developed as early as 1919. Analog speech scrambless basically modify the speech signal by operating in amplitude, time and frequency domain. For example, a relatively elementary technique provides speech scrambling by inversion of the audio frequency band. A major disadvantage of speech scrambling systems of this type that they are quite readily susceptible to being intercepted and unscrambled by other than the desired receiver. While simple and inexpensive, the system suffers from the further disadvantage that it is substantially unsuited to implementation in a form providing more than a few scrambling codes. A related but slightly more versatile and hence more complicated scrambling system which has been utilized in the prior art divides the voice communications channel into a plurality of subchannels each including a portion of the frequency range of the entire voice communication channel. Each of these narrow frequency bands may be individually inverted and heterodyned with a fixed reference signal or plurality of fixed reference signals and then recombined in a preselected order to provide a scrambled signal of greater complexity and therefore less susceptible to being intercepted and unscrambled by other than the desired receiving station. Further, by merely changing the order of recombination of the several speech sub-bands, a speech scrambler of the type described may provide a number of different and distinct scrambling codes by merely reprogramming rather than restructuring the device. More complicated techniques were developed in order to make the reverse process more complex for any unintended recipient of the signal. The permutation of speech segment [63] and transform based analog speech scramblers [64] can be considered as the major analog speech scramblers. Sophisticated techniques for their cryptanalysis also exist like [62, 65, 66, 67]. With the advancement in digital technology speech is now considered as an ordinary stream of bits comprising of ones and zeros. All the properties of speech which were exploited for cryptanalysis in case of analog encryption techniques are no more valid in the digital domain. The statistical properties of digitized speech, however, are studied and discussed from crytpanalysis point of view for the first time in this thesis.

3.3.2 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. Following are the most prominent of the works discussed in the literature regarding use of HMMs for cryptanalysis.

3.3.2.1 Fast Dictionary Attacks on Computer Passwords

Human memorable passwords are the basis of first level computer security. A. Narayanan and V. Shamtikov used hidden Markov models for improving fast dictionary attacks on human memorable passwords [68]. The basis of this attack is that the distribution of letters in the human memorable passwords is similar to the native language of the password user. It is shown that using standard Markovian modeling techniques the password search space can be drastically reduced. The search speed is further augmented by time memory trade off. The algorithm presented in [68] has been evaluated on the real world human memorable password hashes and successfully recovered 67.6 percent of the passwords using a search space of $2x10^9$. Our previous argument that all the techniques involving use of HMM for cryptanalysis presented so far relate to text based data is further reasserted by the use of HMMs for dictionary attacks on passwords.

3.3.2.2 Timing Attacks on Secure Shell

Secure Shell (SSH) is designed to provide secure communication between two hosts in a network. In SSH interactive mode every individual keystroke is sent to the remote machine in a separate packet immediately after the key is pressed. This immediate transmission of keystroke information can be captured and exploited to gain timing information of users' typing. In [69] statistical techniques using HMMs are applied for modeling the timing information of keystrokes to reveal key sequences in a SSH session. If the attacker can infer what the user is typing from the inter keystroke timings then he can learn what users type in a SSH session from the packet arrival times. The technique using HMMs presented in [69] can reduce the cost of password cracking by a factor of 50 for passwords of length upto eight characters and hence speed up exhaustive search dramatically. The attack presented in [69] is not only applicable to SSH but can be effectively applied to all such protocols which deal with interactive traffic. We see here agin that HMMs are applied in cryptanalysis of text based data.

3.3.2.3 Deciphering Substitution Cipher

Language models have been effectively used in deciphering of simple substitution ciphers. In [70] D. Lee proposed a solution to the substitution deciphering problem using hidden Markov models and demonstrated its application to symbolically compressed document processing. It has been point out the abstraction of state transitions from symbol observations in an HMM is analogous to the separation of a Markov language source model and a substitution enciphering process. Therefore, an HMM properly initialized with the statistics of a source language can be used to find the most likely character mappings for a given cipher passage. Again the application of HMMs to deciphering of substitution ciphers deal with use of HMMs for cryptanalysis of text based data.

3.3.2.4 Modeling Keyboard Acoustic Emanations

Emanations produced by electronic devices have long been studied for the purpose of extracting some hidden information about the emanating device. These emanations may be in the form of electromagnetic waves or may be sound waves, these render some useful information about their source. It was only recently that acoustic amanations of keyboard were captured to find what was being typed from the sound produced by different keys [71]. L. Zhuang, F. Zhou and J. D. Tygar modeled the keyboard acoustic emanations as HMMs [72]. In this case a 10 minute sound recording of a user typing English is taken and using hidden Markov models upto 96 percent of the typed keys are successfully recovered. The attack uses a combination of standard machine learning and speech recognition techniques, including Cepstrum features, Hidden Markov Models, linear classification and feedback-based incremental learning. In this case HMM based speech recognition techniques are applied but not to identify speech but to identify the key pressed by a user from the sound which it produces.

3.3.2.5 Counter Measure against Side Channel Attacks

Side channel attacks are recently introduced in the domain of crytpanalysis. These attacks in the from of timing attacks, power analysis attacks and differential power analysis attacks have shown encouraging results for breaking into the implementations of otherwise quite secure algorithms. Randomized countermeasure is one of the techniques for thwarting side channel attacks. such attacks. In [73], C. Karlof and D. Wagner used HMMs for breaking the randomized countermeasures against side channel cryptanalysis by modeling these counter measures as Input Driven Hidden Markov Models (IDHMMs) - the concept also introduced here for the first time. IDHMM is a generalization of HMM that provides a powerful and unified cryptanalytic framework for analyzing countermeasures whose operational behavior can be modeled by a probabilistic finite state machine. The use of HMMs in this case differentiate this attack from other attacks on randomized countermeasure in two ways. Firstly it makes the attack generalized over a broader class of countermeasures and secondly it is applicable even if the side channel is noisy.

3.3.2.6 Cryptanalysis of Two Time Pad of Text Based Data

Finally the most relevant work to ours is that of Mason et al [37] who used Markov models to solve the two time pad problem for the stream ciphered uncompressed text based data. Our algorithm for cryptanalysis differs from this work in two aspects. Firstly, the exploitation presented in [37] will only work on text based data and will not be applicable if the underlying plaintext is speech as it uses smoothed n-gram language models in which the transitional probabilities are calculated from the probability of a specific character followed by another character in the English language. Secondly, it assumes the underlying plaintext to have some probability distributions which is not uniformly i.e. if compression is applied over the plaintext before encryption which make the data uniformly distributed then the attack will fail. Our technique of exploitation on the speech signal works even in the highly compressed encoding techniques applied over the speech signal before being stream ciphered.

It is worth mentioning here that the works involving cryptanalysis with the aid of HMMs presented so far either do not relate to two time pad cryptanalysis or pertain only to text based data. 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.

3.4 Proposed Approach

Before discussing the concept behind our approach for exploiting keystream reuse in stream ciphered digitized speech signals, we first elaborate and justify our assumptions.

3.4.1 Assumptions

We assume that the cryptanalyst knows before launching an attack, the details of the speech digitization and encoding before being stream ciphered. As an a priori knowledge the cryptanalyst must know whether the audio data which he targets is waveform encoded or parameter encoded. This assumption becomes realistic from the fact that the audio encodings follow some standards and by merely knowing the standard reference, the bit level details can be easily accessed as these details are publicly available. For example, in the waveform coding we may have ITU-T G. 711 [3], the bit level details of which are publicly available. Had these details not been available then even the plaintext speech without encryption would have not been possible to be decoded. We also assume that the cryptanalyst has a priori knowledge of the language and genre of the underlying speech signal. For example, we may like to model military telephonic conversations, corporate discussions, and informal chat over mobile telephones, etc. In case of VENONA project [61], the NSA knew before hand that the said link carried Soviet military and diplomatic communication. Moreover, keeping in view the Kerckhoffs principle re-asserted by Claude Shannon as the enemy knows the system, the language and encoding details of the underlying speech signals can be rightly assumed to be known to the cryptanalyst before hand.

3.4.2 The Concept

Our method of cryptanalyzing the speech signals being encrypted with the same keystream is based on the hidden Markov model based speech recognition techniques. All modern speech recognition tools use this technique because of its robustness, flexibility and efficiency [57]. The three basic questions with respect to the HMMs and their solutions as regards to speech recognition are effectively utilized with some modification in our case. The three basic problems of interest that are to be solved for the model to be useful as regards to the cryptanalysis of speech signals encrypted with the same key are:

3.4.2.1 Finding Probability of Observation Given a Model

Given an observation sequence O (XORed ciphered speech waveforms in our case) and a model where is the initial probability of states (XORed phonemes in our case), 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 i.e. ? The solution to this problem allows us to choose the model which best matches the observation sequence which in our case would be a sequence of XORed speech samples.

3.4.2.2 Finding Internal States Given Model and Observation

Given the observation sequence O of XORed speech waveforms and the model , 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? This is the problem in which we try to find the hidden part of the model i.e. to find the correct state sequence. In this case we have to impose certain optimality criterion like the sequence with maximum log probability may be selected.

3.4.2.3 Adjusting Model Parameters Given Observation

How do we adjust the model parameters to maximize the probability of the observation sequences of XORed speech waveforms given the model ? This is the training part of the models and on one hand is the most crucial part in the sequence of events but on the other hand gives a lot of flexibility to the cryptanalyst thus empowering him to adjust his model for all sorts of varying situations like different languages, accents, different speech coding and compression techniques and even different noisy conditions. This allows the crypanalyst to create best models for real phenomena.

All the abovementioned problems and their efficient mathematical solutions have a very rich literature with respect to speech recognition [57]. In the conventional speech recognition techniques, the hidden Markov models are trained for complete words in case of isolated word recognizers and for phonemes 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 n2. 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 is to train the HMMs for the exclusive ORed pairs of all the possible phonemes in the language under test. In this case, since the number of phonemes is relatively very small as compared to the number of possible words, the increase in computational complexity from n to n2 does not become unachievable. 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 2500 (502) which are not high as regards to the computational resources available to a normal user these days. Using this approach would not require any major modification in the conventional phoneme based speech recognition procedure. In the pre-computation phase which comprises of selection and training of HMMs, we will first list down all the possible phonemes in the language and then we will pair each phoneme with every other phoneme including itself in the list. We will then assign each HMM to each phoneme pair and then train it with the training data selected on the basis of the a priori knowledge about the language and encoding mechanism of the speech signal. Once the HMMs corresponding to phoneme pairs are fully trained then these can be used for the identification of phoneme pairs in a given sequence of XORed speech samples in a process similar to the decoding part of a conventional HMM based ASR system. The decoded phoneme pairs are first separated into two distinct groups and then the phonemes within a group can then be and combined to form different words and sentences. For both these steps, help can be taken from the semantics and syntactic rules of the language.

3.5 Implementation of Proposed Approach

The implementation part of our attack involves a pre computation phase which involves selection and training of the models and then the decoding phase which corresponds to the recognition of sequence of phoneme pairs which gives the highest log probability. 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.

3.5.1 HMM Selection and Training Phase

This phase corresponds to the pre computation part of the speech recognition in which the HMMs are first selected and then trained with the help of speech samples available with respect to each phoneme pair. This 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 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. Figures 3.1, (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 [74]. 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 in the form of phoneme pairs. The + sign in the figure corresponds to XOR. 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 (27x26) 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. For recordings, transcription and speech recognition we used the HTK [75] which is a toolkit based on C language for analysis of hidden Markov models particularly for machine recognition of speech. The 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 figure3.4. Since speech recognition equipment cannot process waveforms directly, these are to be converted into more compact form. The configuration we used for the speech recognition was based on Mel Frequency Cepstral Coefficients (MFCC) [76] 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 aij 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 from task grammars, 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 to construct the recognition network from the task grammar. HSGen can be used for testing and verification of the recognition network.

3.5.2 Decoding Phase

Once the pre-computation phase has carefully been completed, the decoding process becomes pretty simple and elegant. An input speech signal, which in our case will be the bitwise XOR of two unknown speech waveforms, is first converted to a sequence of n MFCC vectors and is then fed as input to the recognizer. The prospective recognition hypothesis comprises every path from the beginning node to the finishing node which passes through precisely n emitting states. Every one of these paths has a log probability which is actually the sum of 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 between two states are determined

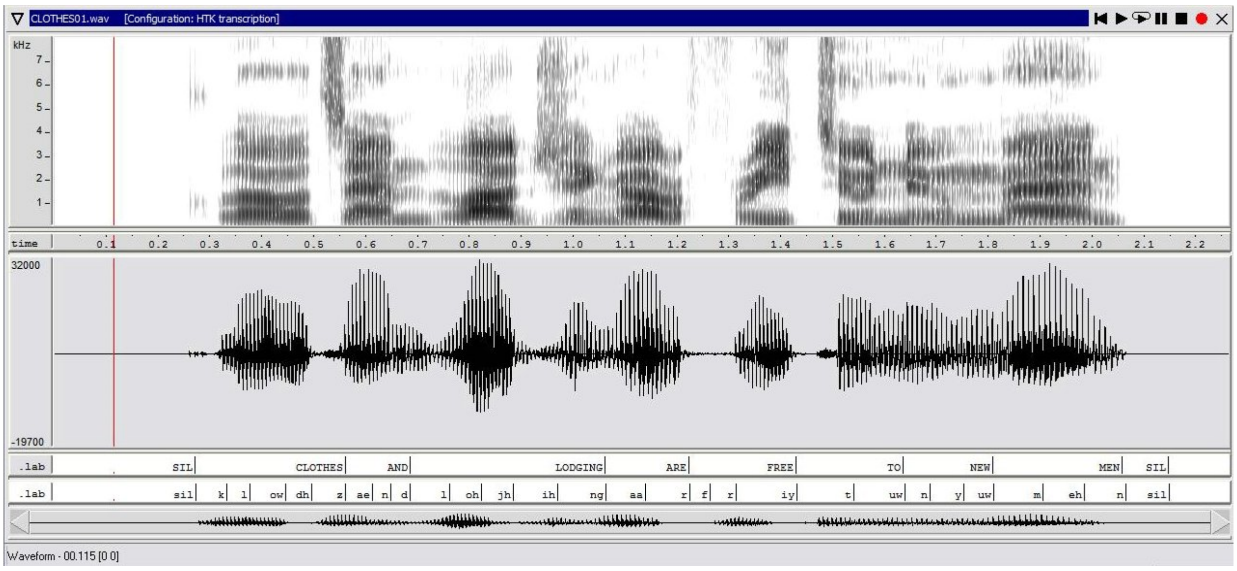
CLOTHES01.wav [Configuration: HTK transcription]
kHz
7
6
5
4
3
2
1
time
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2
32000
-19700
.lab SIL CLOTHES AND LODGING ARE FREE TO NEW MEN SIL
.lab sil k l ow dh z ae n d l oh jh ih ng aa r f r iy t uw n y uw m eh n sil
Waveform - 00.115 [0 0]

Figure 3.1: Spectrogram and waveform along with transcription of the sentence: ***Clothes and lodging are free to new men***

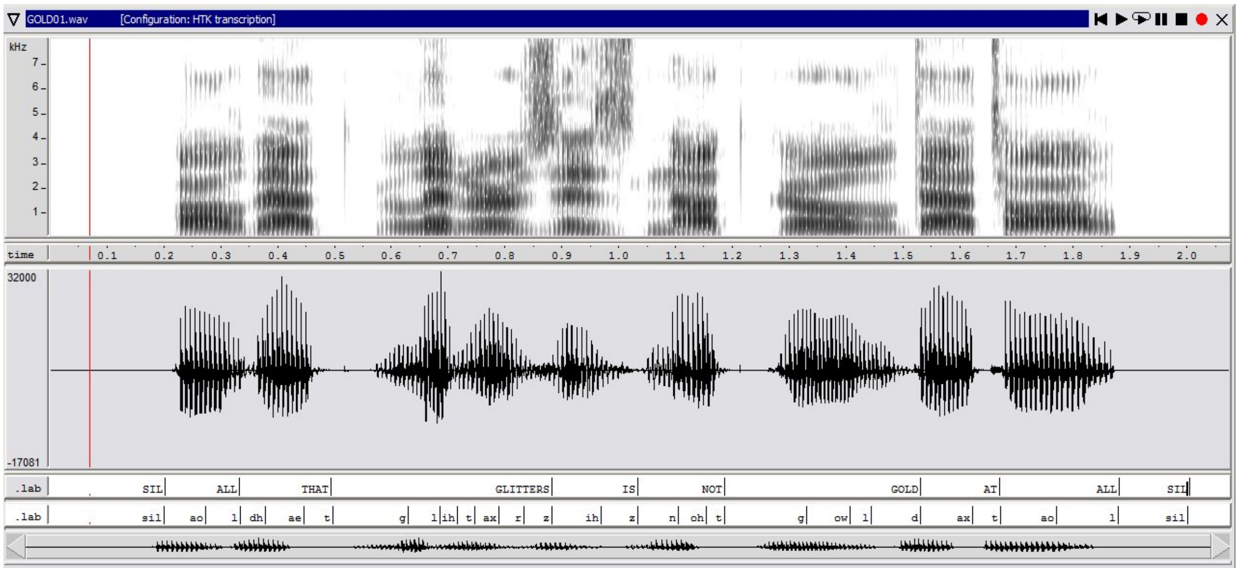
GOLD01.wav [Configuration: HTK transcription]
kHz
7
6
5
4
3
2
1
time
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
32000
-17081
.lab SIL ALL THAT GLITTERS IS NOT GOLD AT ALL SIL
.lab sil ao l dh ae t g l ih t ax r z ih z n oh t g ow l d ax t ao l sil

Figure 3.2: Spectrogram and waveform along with transcription of the sentence: ***All that glitters is not gold at all***

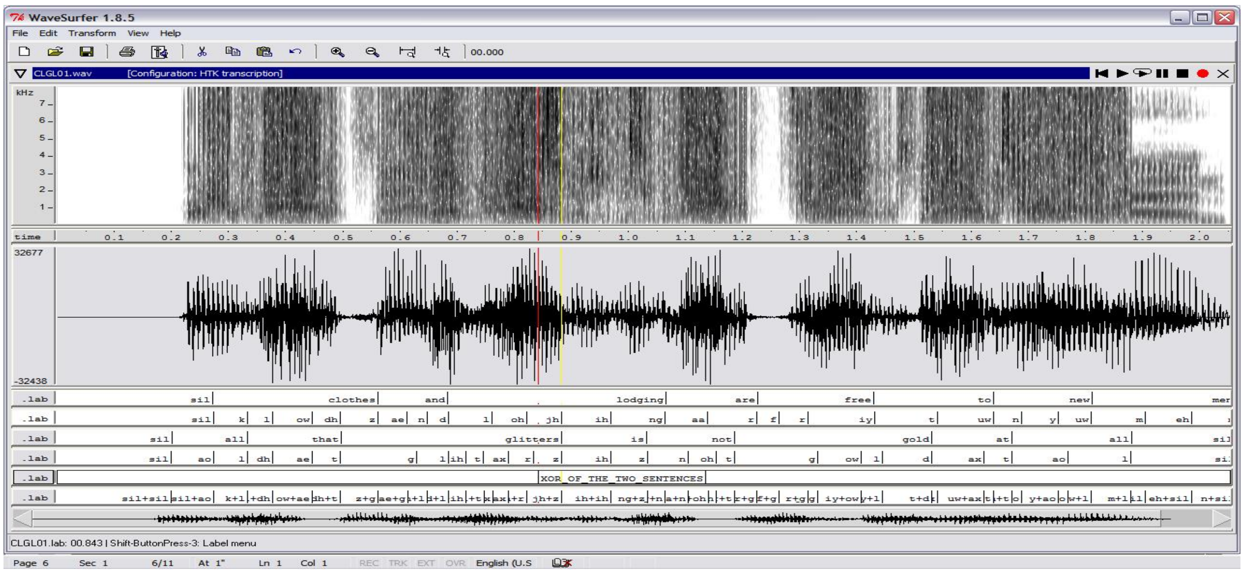
WaveSurfer 1.8.5
File Edit Transform View Help
00.000
CLGL01.wav [Configuration: HTK transcription]
kHz
time
32677
-32438
.lab sil clothes and lodging are free to new mer
.lab sil k l ow dh z ae n d l oh jh ih ng aa r f r iy t uw n y uw m eh
.lab sil all that glitters is not gold at all sil
.lab sil ao l dh ae t g l ih t ax r z ih z n oh t g ow l d ax t ao l sil
.lab XOR_OF_THE_TWO_SENTENCES
.lab sil+sil sil+ao k+l +dh ow+ae dh+t z+g ae+g +l d+l ih +t x ax +r jh+z ih+ih ng+z +n a+n +oh h +t r+g f+g r+g g iy+ow y+l t+d uw+ax t +t o y+ao o w+l m+l il eh+sil n+sil
CLGL01.lab: 00.843 | Shift-ButtonPress-3: Label menu
Page 6 Sec 1 6/11 At 1" Ln 1 Col 1 REC TRK EXT OVR English (U.S

Figure 3.3: Spectrogram and waveform along with transcription of XOR of the sentences

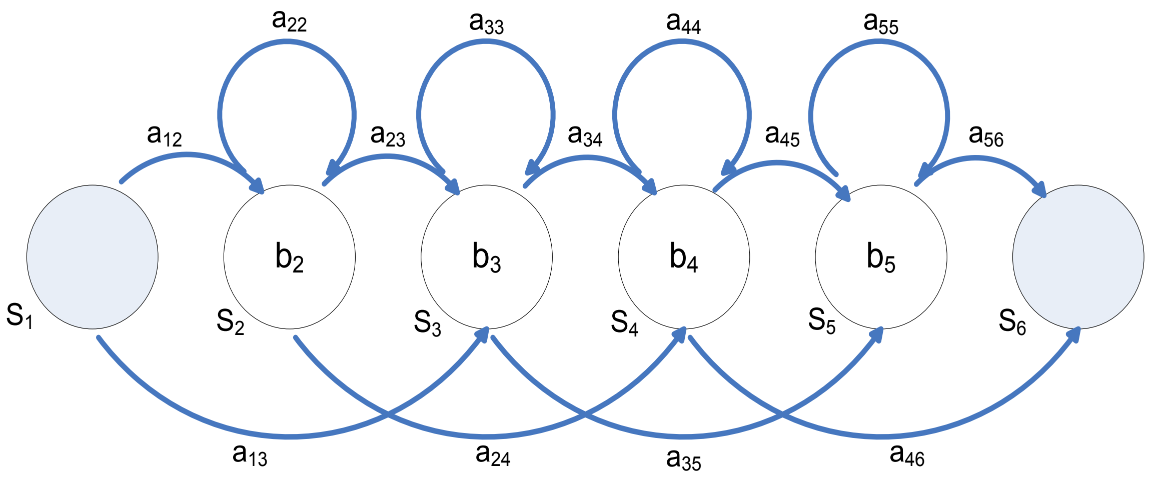

Figure 3.4: Basic Topology of HMM

```
/* 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 ] )
```

Figure 3.5: Task Grammar for the Recognition Network

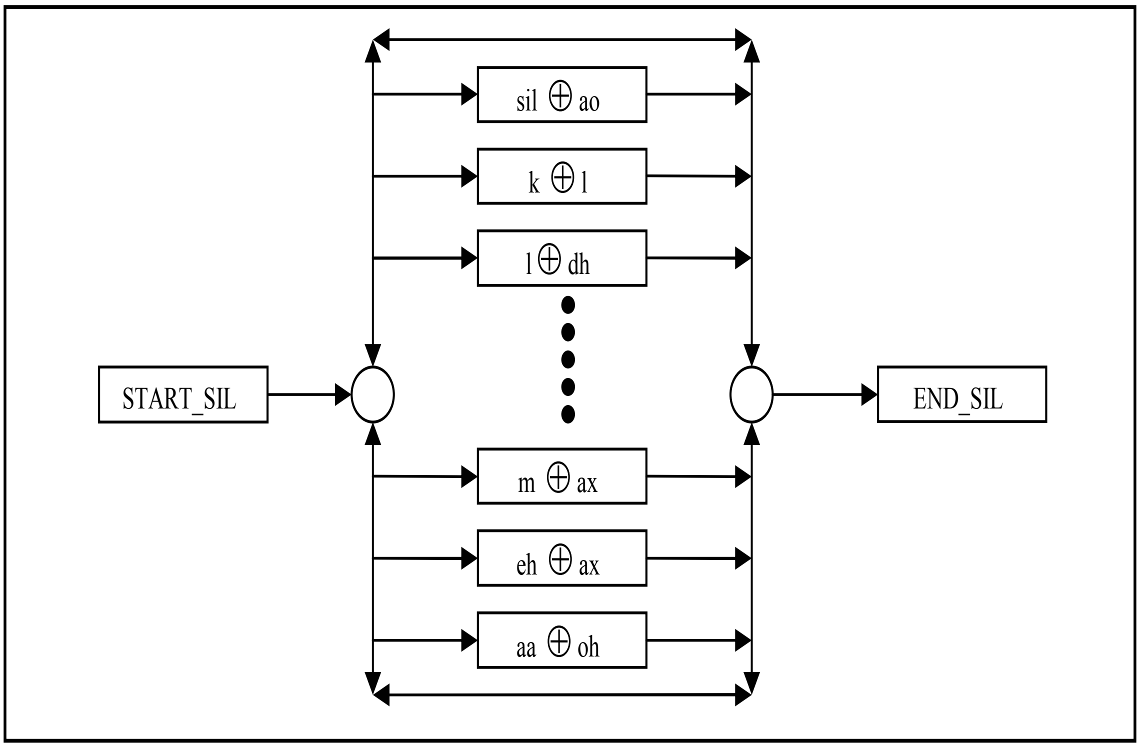

Figure 3.6: Recognition Network

from the model parameters $(a_i j)$. The transition probabilities between any two models are kept as constant whereas the transition between two words words are determined by the language models in case of large networks. upon the threshold set a list of the probable paths are listed by the decoder. A Token Passing Algorithm [75] is used to find these paths. A token is placed in every possible start node at time 0. At each ach time step, tokens are propagated along connected paths which stop whenever an emitting HMM state is reached. In case the out going paths from a node are more than one, all possible paths are explored in parallel by copying the token in all paths. 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 beam width for pruning is set too small then it may result in a search error by pruning the actual recognition path before its token reaches the end of the observation. 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 [75]. The initial experiments which we performed gave us up to 80 percent correct recognition of phoneme pairs obtained though HResults tool of HTK as shown in Fig. 5.

================== 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]

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

Figure 3.6: HTK Recognition Performance

3.5.3 Detailed Experimentation

After getting encouraging results from the experiment in the controlled environment, we then tested our technique on actual speech files. For this purpose, we selected the Switchboard Corpus [77] which is a collection of telephone bandwidth conversational speech data collected from T1 Lines. The speech files are fully transcribed. The reason for this selection was to simulate the situation of eavesdropped encrypted communication of waveform encoded speech. In order to simulate the keystream reuse scenario, we selected 256 speech files and XORed them with each other. The acoustical events were modeled with 588 HMMs with each HMM corresponding to one pair of phonemes. Since all the possible phonemes do not occur always hence the actual number of phoneme pairs (588) is quite less than the total possible number (2500). As discussed earlier, speech recognition tools cannot process speech waveforms directly. Different acoustic feature representations were used for recognition purposes. For all the representations, we used frame length of 25 milliseconds with 10 milliseconds frame periodicity. The parameters which were to be estimated for each HMM during the training phase were transitional probabilities aij 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). The different acoustic features extracted for recognition purposes include linear predictive coefficients, linear predictive reflection coefficients, linear predictive Cepstral coefficients and Mel frequency Cepstral coefficients along with delta and reflection coefficients. For testing purpose, we selected

Table 3.2: Recognition Accuracies of Different Acoustic Features

SNo	Feature Extraction Mechanism	Recognition Accuracy (%)	
		Training Data Test Files	Arbitrary Test Files
1.	Linear Predictive Coefficients	65.93	29.51
2.	Linear Predictive Reflection Coefficients	69.06	28.61
3.	Linear Predictive Cepstral Coefficients	72.09	34.62
4.	Mel Frequency Cepstral Coefficients (MFCC)	74.72	40.73
5.	Linear Predictive Cepstral + Delta Coefficients	77.15	37.81
6.	Mel Frequency Cepstral + Delta + Acceleration Coef.	79.96	59.09

twenty different files from the Switchboard corpus not included in the training data, XORed these to get ten files. As an initial test we also selected ten different files from the training data and fed these to the recognizer. The experimental results with respect to test files selected from the training data as well as arbitrary test files are depicted in Table I. The best accuracy results were presented by the Mel Frequency Cepstral Coefficients (MFCC) with delta and acceleration coefficients for both the test file categories.

3.5.4 Complexity Analysis

For the complexity of the training phase, the increase in the number of models to be trained is from n to n2 at the most, if we pair each phoneme with every other phoneme and of course itself. For the decoding phase, the number of phonemes which are to be checked at each stage of the decoding path is also increased from n to n2 and the number of possible paths increases from n2 at each stage to n4 at each stage. Hence we may expect an exponential increase in the decoding from the conventional speech recognition. One way to reduce the number of iterations at each stage is to carry out beam search in which the width of the beam should be carefully selected in order to avoid pruning of useful paths at the early stage of the recognition network. Fortunately, HTK supports the beam viterbi search approach and hence can be employed in our case with no major modification in the decoding phase.

3.6 Summary

In this paper, we presented that how the keystream reuse problem of stream ciphers can be exploited in case of waveform encoded speech signals. Prior to this work, one could safely reuse keystreams in case the underlying plaintext data was speech as all exploitation techniques presented before this transaction assumed the underlying plaintext to be uncompressed text-based data encoded through conventional encoding techniques such as ASCII coding. We have shown that the conventional speech recognition techniques can be adapted to be used for cryptanalysis of two time pads in case of stream ciphered digitized speech signals. HTK and other automatic speech recognition (ASR) tools can be effectively modified for this purpose. These speech recognition tools work on the concept of context-dependent tied-state multi-mixture tri-phones [75] which make the performance of the recognizer more flexible and robust. The use of tri-phones in the keystream reuse scenario needs to be looked into in the future assignments. The parameter encoded and compressed speech signals in the keystream reuse situation can also be looked into as a future work.

Chapter 4

CRYPTANALYSIS OF TWO TIME PADS OF PARAMETER ENCODED SPEECH

4.1 Introduction

The perfect security of the one time pad is based on the premise that the keystream is as long as the plaintext and is never reused [36]. The keystream reuse problem from the speech signal point of view was discussed for the first time by us in chapter 2 where we used hidden Markov model based speech recognition techniques for this purpose. The techniques presented in chapter 1 can be best applied on waveform encoded speech and does not produce very encouraging results on parameter and hybrid encoded speech signal in the two time pad situation. Since waveform coding has now been replaced by modern parameter and hybrid encoding techniques. In this chapter we address this problem viz a viz the modern speech encoding and compression mechanisms. Digitization, compression and encryption of speech have been important areas of digital communications. Encryption schemes particularly designed for speech starting from the old aged analog speech inverters to the modern aged partial speech encryption techniques have been the focus of security professionals since long. With the advancement in the speech digitization and compression techniques such as low rate Vocoders which use parameter encoding, the speech signal is now treated as an ordinary data stream of bits as far as encryption is concerned. The properties of the digitized speech signals exploited by the low bit rate selective speech encryptor [78], speech recognition equipment especially in the distributed speech recognition [50] and automatic transcription of telephone conversations [51] have encouraged us to look at their characteristics from the cryptanalytic point of view in the keystream reuse scenario.

In case of parameter encoding speech is represented as a source filter model in which the parameters of the model alongwith excitation information in the form of voiced/unvoiced signals are used for digital representation of speech. In hybrid coding, which has become the most popular in modern speech coding, the excitation information is not only segregated as voiced or unvoiced signal but the details of excitation information such as pitch, pulses position/signs and gains are used for its representation. The most common coding technique in this domain is Code Excited Linear Prediction (CELP), which is the most widely used speech coding algorithm. It is for this reason that we have selected a particular implementation of CELP for our attack on parameter/hybrid encoded speech. Different standard implementations of the CELP algorithms are available and are in use in most of the modern speech processing applications. FS1016 CELP at 4.8 kbit/s [79], ITU-T G-723.1 Algebraic CELP at 5.3/6.3 kbit/s [80], ETSI GSM AMR Algebraic CELP at 4.75 12.2 kbits/s [81], ITU-T G-728 Low Delay CELP at 16kbits/s [82], ITU-T G.729 , Conjugate Structure Algebraic CELP at 8 kbits/s [83] are a few such examples. Our implementations will not only address all these standard variations of CELP coded speech, but our technique of attack can also be adapted for other parameter encoding standards, such as LPC 10 E Algorithm by the US Department of Defence [84], Regular Pulse Exited Long Term Prediction (RPE-LTP) used in GSM [85],etc. We will again use the concepts of hidden Markov models combined with some dynamic programming search techniques but from a perspective which is different from the one presented in Chapter 2. We will first devise a mechanism for creating bi-gram models of the parameters obtained after speech encoding. Using these statistical models for each spectral and excitation parameter we will try to find the individual plain text speech signals from their XOR.

4.2 Proposed Approach

Our method of cryptanalyzing the speech signals being stream ciphered with the same key is based on the type of encoding applied to the plain speech before encryption i.e. we assume that the cryptanalyst knows before launching an attack, the details of the speech digitization and encoding before being stream ciphered. This assumption is not unrealistic as most modern speech processing equipment generally use standard encoding procedure as discussed in section 2.1. Moreover, the bit level details of these implementations are publicly available. Had these details not being available then even the plain text speech without encryption would not be possible to be decoded. Moreover, keeping in view the Kerckhoffs principle re-asserted by Claude Shannon as the enemy knows the system, our assumptions do not seem unrealistic. Our attack on the two time pad is based on Markov models and dynamic programming algorithms i.e. we will first try to model the speech we want to cryptanalyze by using the a priori information regarding the language and genre (accent, context, topic etc) of the speech and then the details of the encoding procedure being applied on the plain speech. For example, we may like to model military telephonic conversations, corporate discussions, informal chat over mobile telephones, etc. We will have to first collect a corpus of speech files similar to the ones in genre which we want to attack. We will then try to encode these with the encoding mechanism we want to attack. Then we will create models for each parameter and then train our models on the encoded speech parameters. Since this can be done off line and is required to be done once for a particular genre of speech, hence we can afford to have a high computational and memory complexity. Once we have the trained models available, we can then use dynamic programming techniques like the Viterbi search to find the most probable pair of p and q given x such that . As discussed earlier, Code Excited Linear Predictive (CELP) coding is the most popular technique of speech encoding.

Hence, we will be tackling the CELP coded speech signals. The same concepts can be applied to all other speech encoding techniques in vogue. As discussed and justified earlier, we assume that the frame structures and bit allocations to different parameters in the CELP coded frame like Line Spectral Frequencies (LSFs)/Log Area Ratios (LARs), Pitch delay, Pitch gain, Excitation pulse position, gain and signs, etc being transmitted are known to the cryptanalyst. But instead of having these parameters independently he has two different speech signal parameters (bitwise) XORed together as to simulate the two time pad situation. The job of the cryptanalyst is to find p and q from . If we know the probability distributions of p and q as Pr1 and Pr2 respectively before hand, we can estimate the joint probability of p and q which would be simply Pr1.Pr2 since both p and q are independently selected.

4.2.1 Problem Definition

We can define our problem statement as: *Given an observation sequence O of finite length l such that , in the form of CELP coded vectors of two different speech signals, find the most probable path through the finite state automaton formed by the intersection of two finite state automata with probability distributions Pr1 and Pr2 respectively. Reading off the labels at each state of the most probable path would give us p and q.*

The challenge over here is how to estimate Pr1 and Pr2. For the text based data, language modeling and Natural Language Processing have a very rich literature and the techniques are well matured. If we are able to adapt those well matured techniques to our situation then our problem of cryptanalysis does not remain intractable. If we analyze each frame of a CELP coded speech signal it comprises of certain spectral and the excitation/residual signal parameters and each spectral and residual parameter is encoded with different number of bits in each standard implementation of parameter encoded speech. It means that each component of the speech vector corresponding to each frame can be considered as an alphabet in a language with the total number of alphabets in that language as 2number of bits representing the alphabet . In all standard implementations of parameter encoded speech, the maximum number of bits representing each component generally does not exceed 10, thereby making the language size (the number of alphabets) to a maximum of 1024. Another important consideration over here is that we have to model each component of the CELP vector separately over substantially long streams of CELP frames. If the probability distribution of the component is uniform then our method of attack will fail. Fortunately, in the practical scenario even in the most compressed speech files the probability distribution of the individual component in the speech frame is not uniform for most of the components. For text based data upto 32-gram models are available, and by having models of higher order the accuracy of the attack can be enhanced. In our case, since the number of alphabets is generally higher than text based data and for decoding the complete frames we will have to consider as many language models as the number of components in a speech frame. Hence having models of the order of 3 and more are generally not feasible and requires enormous computational and memory requirements. The time and memory complexity of our algorithm are also discussed in this chapter. Since we are modeling each component of the frame separately, hence we will have to decode each component separately for the entire speech file. One complete frame will only be available once all the components of all the available speech frames are decoded. For the decoding part we used the Viterbi algorithm based on dynamic programming which finds the most optimal path through the finite state automaton. After finding the most probable sequence of frames of both the speech files the separate speech files can be re synthesized. Since 100 percent recovery of the individual frames is not practically possible but fortunately even an accuracy of about 60 percent at regular intervals also produces legible speech but with degraded quality. Since quality may not be a consideration for the cryptanalyst, legibility of the spoken words serves the purpose in most of the cases. Hence, even if we are able to reconstruct the frames of speech with around 60 percent correct components at some regular intervals over the stream of frames, then our purpose of cryptanalysis is achieved.

4.3 Implementation of Proposed Approach

The implementation part of our algorithm can be divide into two main parts i.e. the pre computation part which includes modeling/training part and the decoding part. The pre computation is required to be done once for a particular genre and encoding technique. It can be done off line using the a priori knowledge about the target of interest.

4.3.1 Parameter Modeling and Training

We have implemented our proposed algorithm on one particular implementation of CELP in which each frame corresponds to 20 ms of speech (160 samples at 8000 samples per second). For the short term prediction, 10 line spectral frequencies (LSFs) are calculated for each frame. In order to encode these line spectral frequencies these are divided into sub vectors of 3, 3 and 4. Each sub vector is represented by 10 bits that in fact correspond to the index of a code book which is basically an array of size 1024x3 entries of LSF coefficients. The second code book is also an array of 1024x3 whereas, the third code book is an array of size 1024x4. For a particular implementation of CELP these code books are fixed and we assume that these code books are known to the cryptanalyst before hand, just like the ASCII codes of different characters in a text based data. Hence the initial 30 bits of the frame represent the 10 line spectral frequencies. For the long term prediction coefficients each frame is further divided into four (in some implementations, two) sub frames. The long term prediction components include information regarding the excitation signal in the form of pitch, significant pulse positions and gain information per sub frame which comprises of 40 samples in our case. The details of all these components for our and different standard implementations per frame are tabulated in Table I. Each frame, generally comprising of 10 to 30 milliseconds of speech, is represented by a vector of 30 to 50 components. The CELP implementation, which we have selected, comprises of a total of 47 components (3 LSF indices, 4 Pitch delays, 4 Pitch gains, 16 pulse positions, 16 pulse signs, and 4 pulse gains) per frame to represent 20 milliseconds of speech. The reason for this selection is that this implementation generally covers almost all the different components which are used for speech frame representation. Looking at the bit allocation table, if we model the index of the 1st LSF vector as a Markov process, then we will have 1024 different states. These 1024 different states can be considered as 1024 alphabets in a language and can be modeled as language modeling is done in case of text based data. Since the number of alphabets in a language are generally less than 1024, hence n-gram models of higher order are possible to be calculated but since the number of alphabets is large in our case we will restrict to bi-gram models (at least in this implementation). Moreover, the bi-gram models have produced much better results as compared to unigram models, n-gram models of higher order, if used in future, are expected to further improve the decoding accuracies.

Since there are 47 different components in one frame of speech we will have to create models for all these 47 components separately as discussed in the succeeding paragraphs. Some of the components have high number of bits assigned to a particular attribute but these can be broken down into smaller parts to keep the size of the alphabet within manageable limits for the purpose of modeling as well as decoding. In order to model these different parameters, we selected 200 files (each comprising of about five minutes duration) from the SWITCHBOARD Corpus [77] of conversational speech collected over the telephone bandwidth. The reason for this selection was to simulate the environment of verbal conversation eavesdropped or wiretapped over a communi-

Table 4.1: Bit Allocations for Various Parameters in Standard and Our Implementations of CELP coded speech

Parameters	Bits Allocation per Frame				
	G.723.1 Low Rate	G.723.1 High Rate	G.729[1]	GSM 6.60	Our
Bit rate	5.3 kbits/s	6.3 kbits/s	8 kbits/s	12.2 kbit/s	9.5 kbits/s
LSF/LAR[2] Indices	24	24	18	38	30
Pitch Delay	18	18	14	30	28
Pitch Gain	48	48	6	16	24
Fixed Code Book Pulse Positions	48	73	26	140	68
Fixed Code Book Pulse Signs	16	22	8	0[3]	16
Fixed Code Book Gain / Grid Index[4]	4	4	8	20	24
Total	158	189	80	244	190

cation channel.

4.3.1.1 Line Spectral Frequency Index Modeling

Since we have 3 components representing the 10 different LSFs and are represented using 30 bits in total, if we model these parameters together then the size of the alphabet would be 230 but by modeling each component separately the complexity is reduced to 210+ 210+210. Moreover, each component has a different probability density function as shown in figures 4.1, 4.2 and 4.3, hence modeling these separately will have a positive impact on the decoding accuracies as well. In our implementation, the index of the 1st LSF vector has a minimum value of 2, maximum value of 1022, mean value of 498.5 and standard deviation as 303.25. For the second index, the minimum value is 0, maximum value is 1023, mean is 518.3 and the standard deviation is 290.3. The third index has a minimum value of 0, maximum value of 1022, mean of 462.5 and a standard deviation of 301.8. Here, instead of the mean we use the mode (i.e. the value with the highest frequency) which may better represent the data, hence the mode for the three indices are 6, 629 and 262 respectively.

4.3.1.2 Pitch Delay Modeling

The pitch delay is represented by 7 bits per sub frame and we have to create four different models for the pitch delays each of 128 alphabets each. Figure 4.4 represents the probability density function of the pitch delay of the four sub frames. The mode for the pitch delay values is 111 for all the four sub frames whereas there is some variation in the means and standard deviation of the pitch delay values of the four sub frames. The mean values are 74.02, 70.5, 70.28 and 75.67 whereas, the standard deviation values are 35.73, 36.4, 37.25 and 36.58 respectively for the four sub frames. We can model these parameters jointly but we modeled them separately to keep the

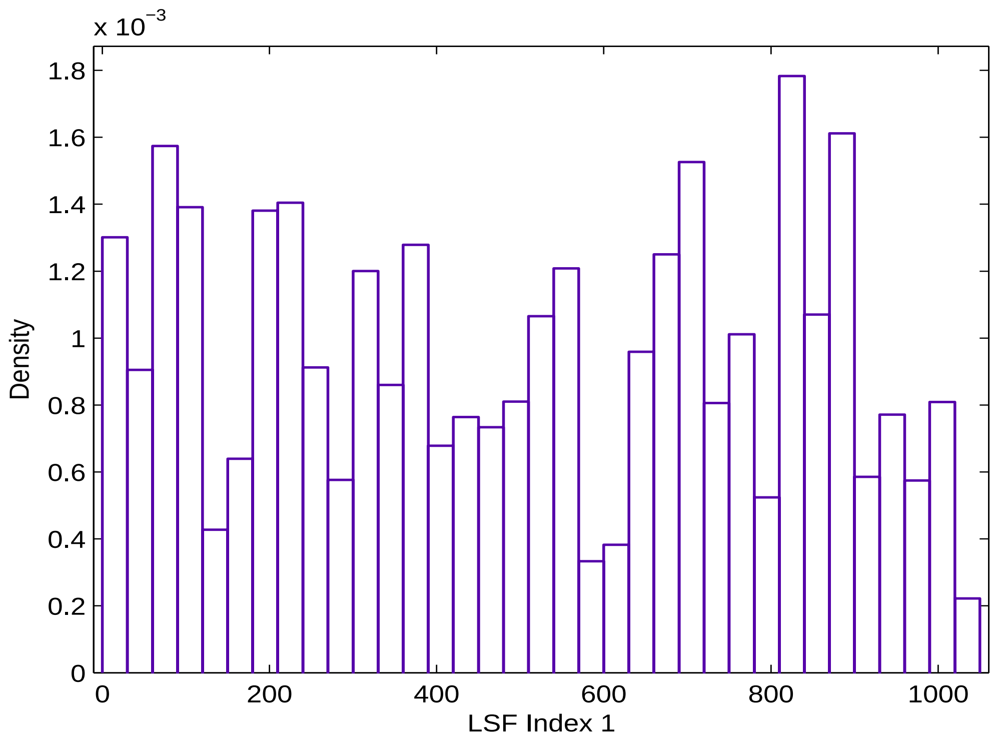

Figure 4.1: Probability Density Function of the Index of the Codebook which Represents LSF1 to LSF3

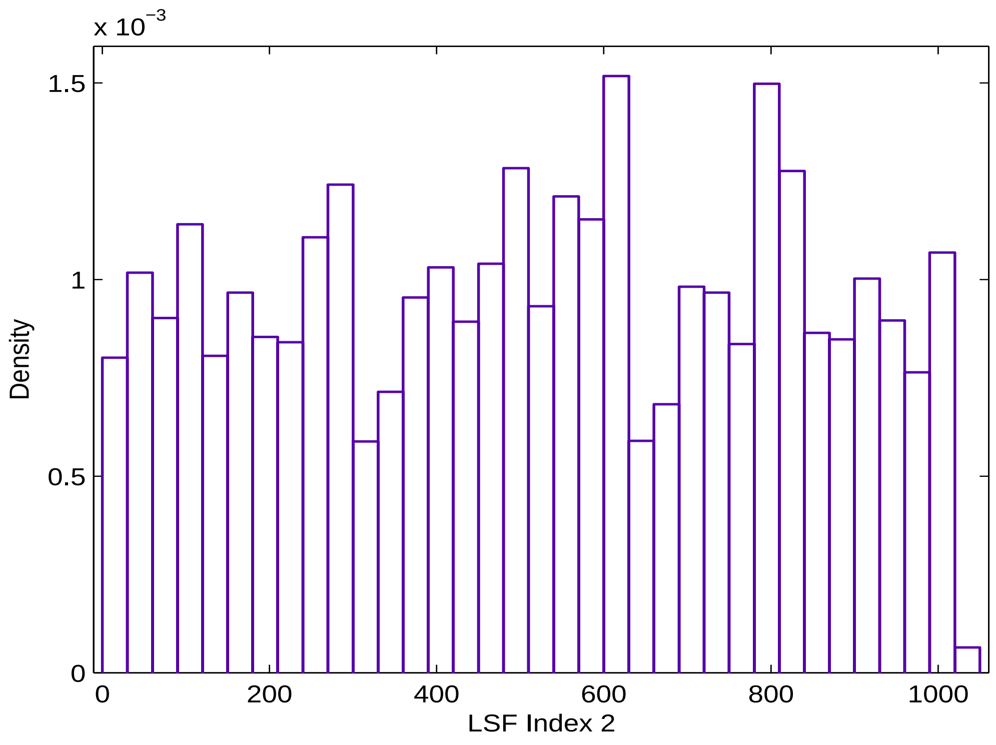

Figure 4.2: Probability Density Function of the Index of the Codebook which Represents LSF4 to LSF6

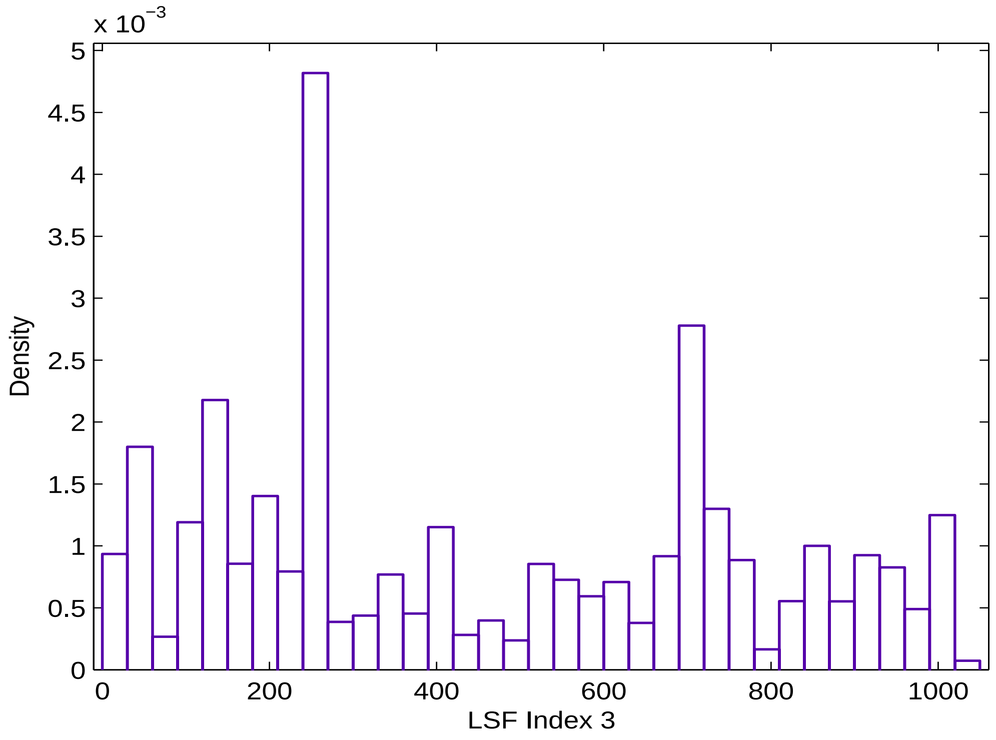

Figure 4.3: Probability Density Function of the Index of the Codebook which Represents LSF7 to LSF10

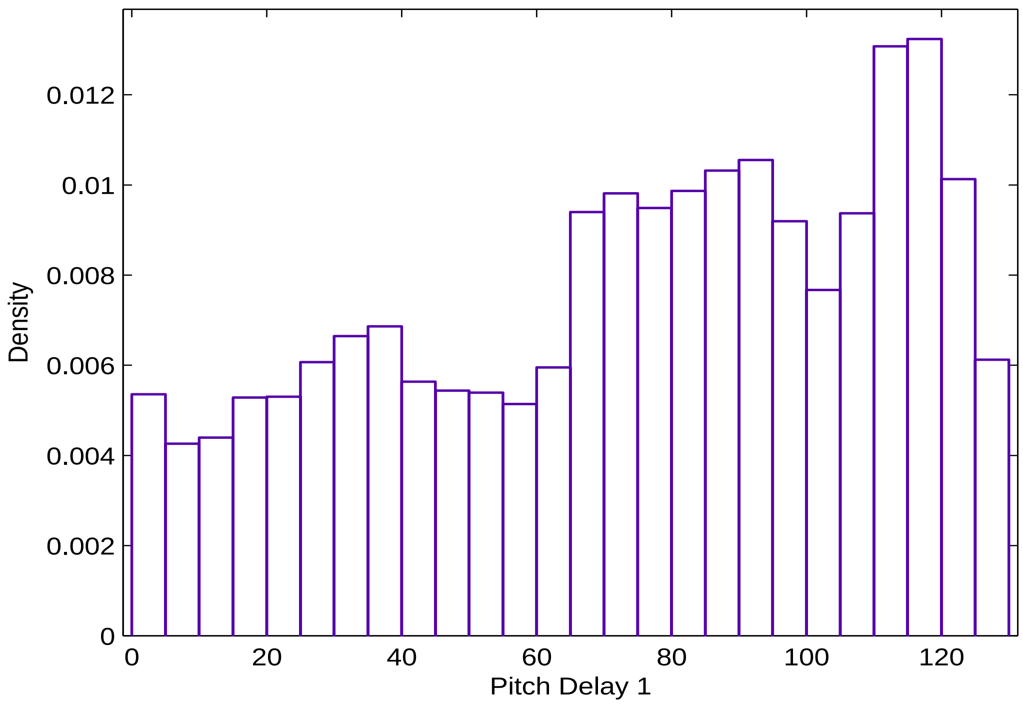

Figure 4.4: Probability Density Function of the Pitch Delay for the four sub frames

size of the alphabet low.

4.3.1.3 Pitch Gain Modeling

Although there are 6 bits dedicated for the pitch gain in our implementation yet mostly the values lie between 49 and 51 as can be seen on the probability density plot in figure 4.5. For the alphabet size we will still consider 64 because the XORed value may lie between 0 and 63. The high frequency value of the pitch gain for all the four sub frames is 51. Similarly the mean value for all the four sub frames remain at 50.8 whereas, the standard deviations vary slightly. The standard deviation values for the four sub frames are 0.935, 0.959, 0.921, and 0.931 respectively. This data can also be modeled jointly.

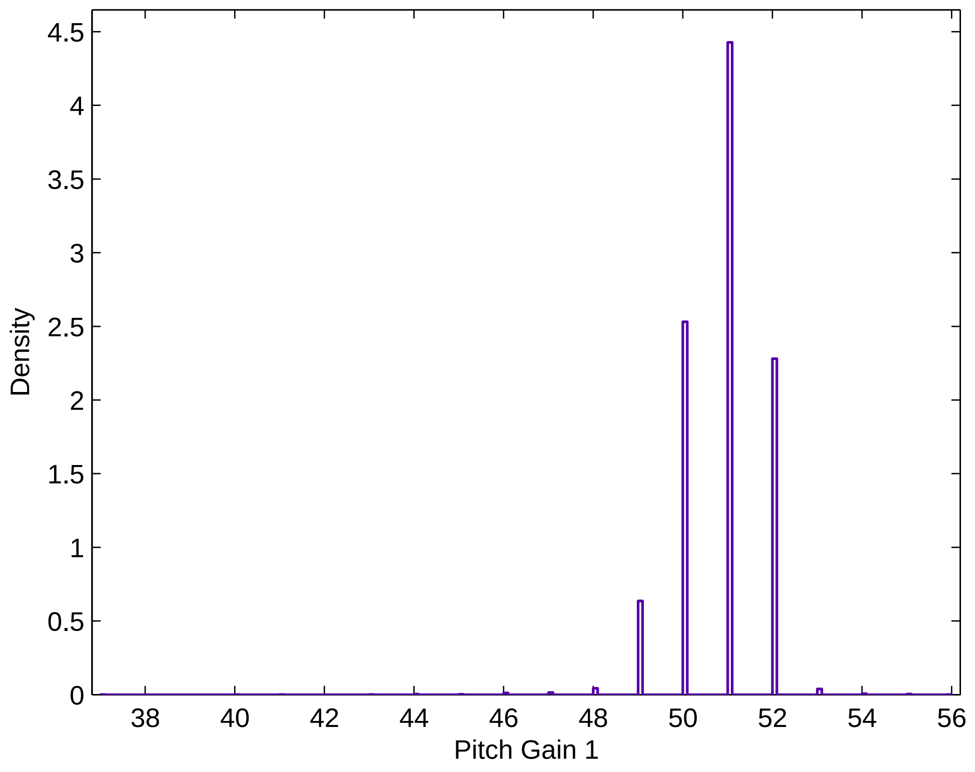

Figure 4.5: Probability Density Function of Pitch Gain for sub frame 1

4.3.1.4 Fixed Code Book Pulse Position Modeling

The fixed code book pulse positions refer to the 4 most significant pulses selected from a sub frame comprising of 40 samples in our implementation of CELP. Hence there are a total of 16 different pulses selected out of 160 samples per frame. These pulse position values lie between 1 and 40 and a total of 17 bits are allocated for its representation per sub frame. All these 16 values are modeled separately. As it is clearly evident from figure 4.6 of the four pulse position in one sub frame that the 1st position value will be close to the earlier values, the second value will lie in the first half, the third value in the second half and the fourth pulse will be selected in the later or higher values. In our implementation the highest frequency value for the first pulse position of a sub frame (pulse position 1, 5, 9 and 13 of the frame) is 1, for the second pulse position in a sub frame (position 2, 6, 10, and 14 of the frame) it is 14, for the third pulse position in the sub frame (position 3, 7, 11 and 15 of the frame) it is 31 and for the last pulse position in the sub frame (position 4, 8, 12 and 16 of the frame) it is 40.

4.3.1.5 Fixed Code Book Pulse Sign Modeling

These refer to the polarity of the pulses selected as the significant pulse positions. Since there are two possibilities for the sign, therefore one bit per sign will suffice for complete representation. The size of the alphabet is just two, and the modeling can even be done collectively for even all the 16 different signs per frame. We have not done it collectively in order to keep our decoding simple as we have modeled each sub component of the 47 component speech vector separately. As is evident from the probability density function of the signs shown in figure 4.7, the distribution of the positive and negative pulses is almost uniform and that is why the accuracy of the signs decoding is very low in our probabilistic model. Presently, we have randomly distributed our

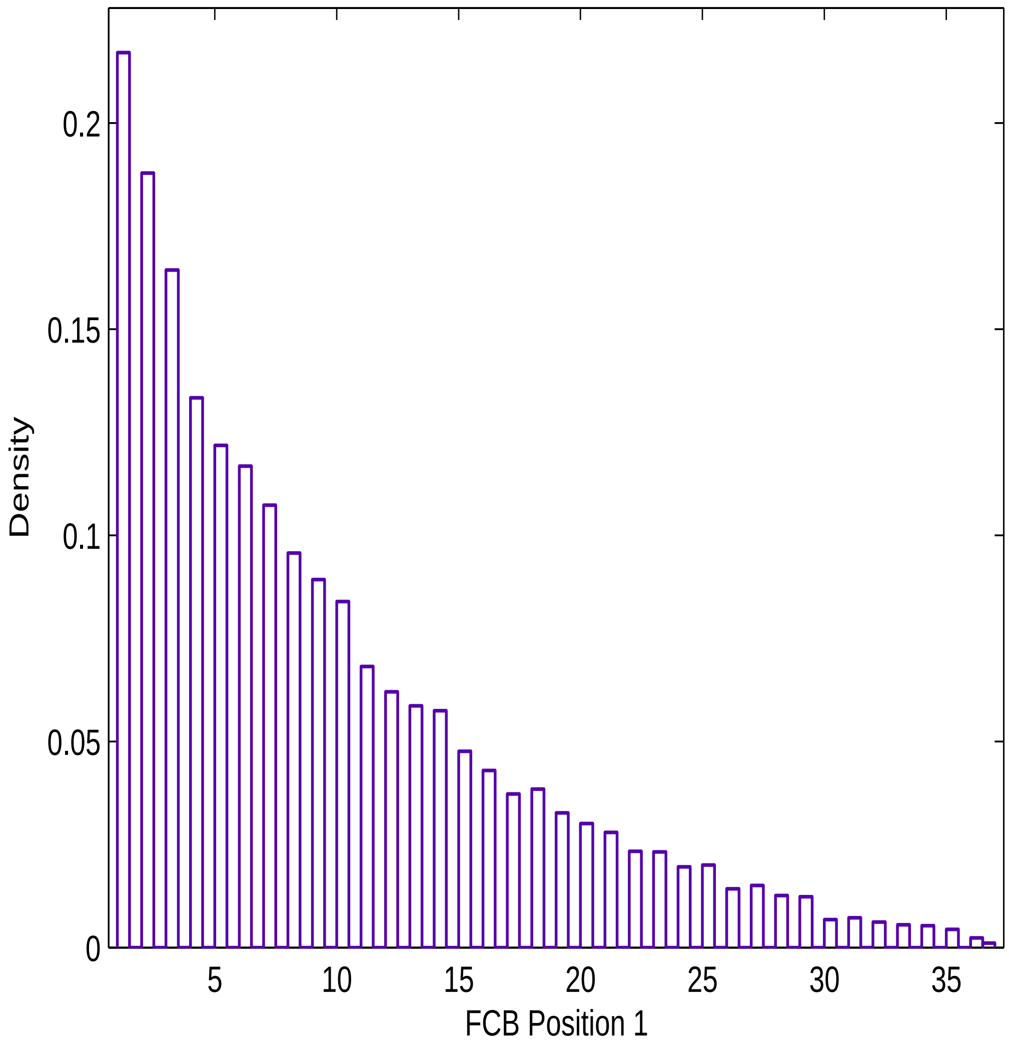

Figure 4.6: Probability Density Function of the 4 Fixed Code Book Pulse Position of sub frame 1

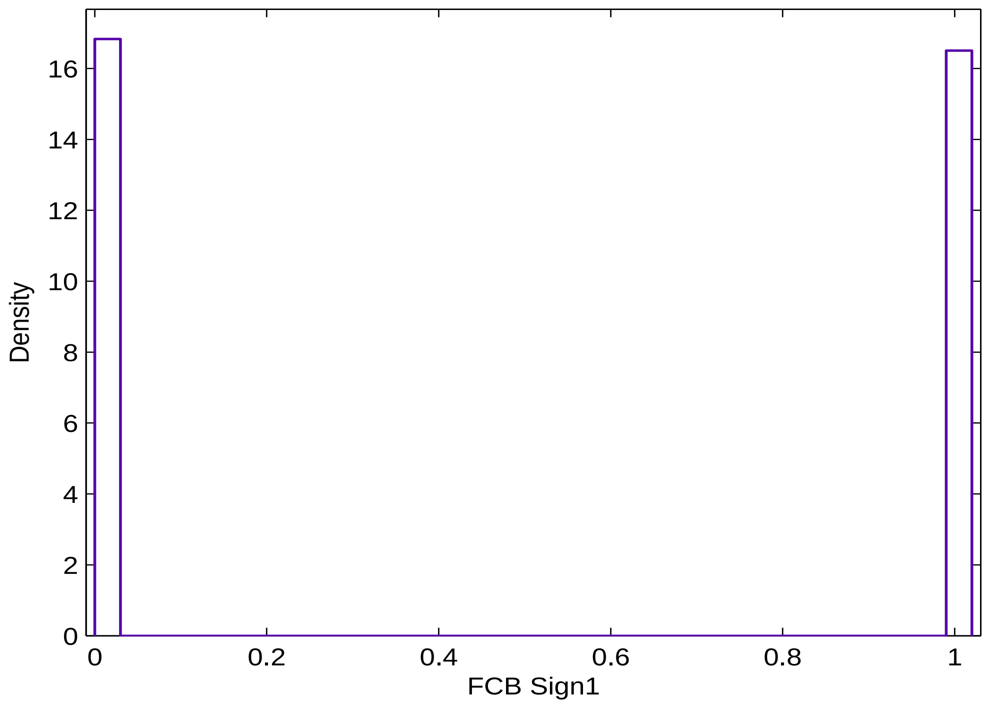

Figure 4.7: Probability Density Function of Fixed Code Book Sign 1 of sub frame 1

signs with uniform distribution over the decoded frames which has served the purpose of speech cryptanalysis, but the decoding of signs may be taken up as a future task and we need to look into modeling these differently from rest of the parameters, to yield better decoding results.

4.3.1.6 Fixed Code Book Gain Modeling

The fixed code book gain refers to the gain of the four pulses per sub frame, thereby having 4 fixed code book values per frame. 6 bits per sub frame are allocated for the four pulses. The probability distributions of all the four sub frames in our training data are alike with the highest frequency value as 20, mean as 28.15 and standard deviation as 13.5 for all the four sub frames. It can be inferred from this that one model can be used to represent the fixed code book gains of all the four sub frames but by careful examination of the probability density plots in fig 4.8, it is evident that there is variation in the bi-gram probabilities of the individual alphabets. Hence the

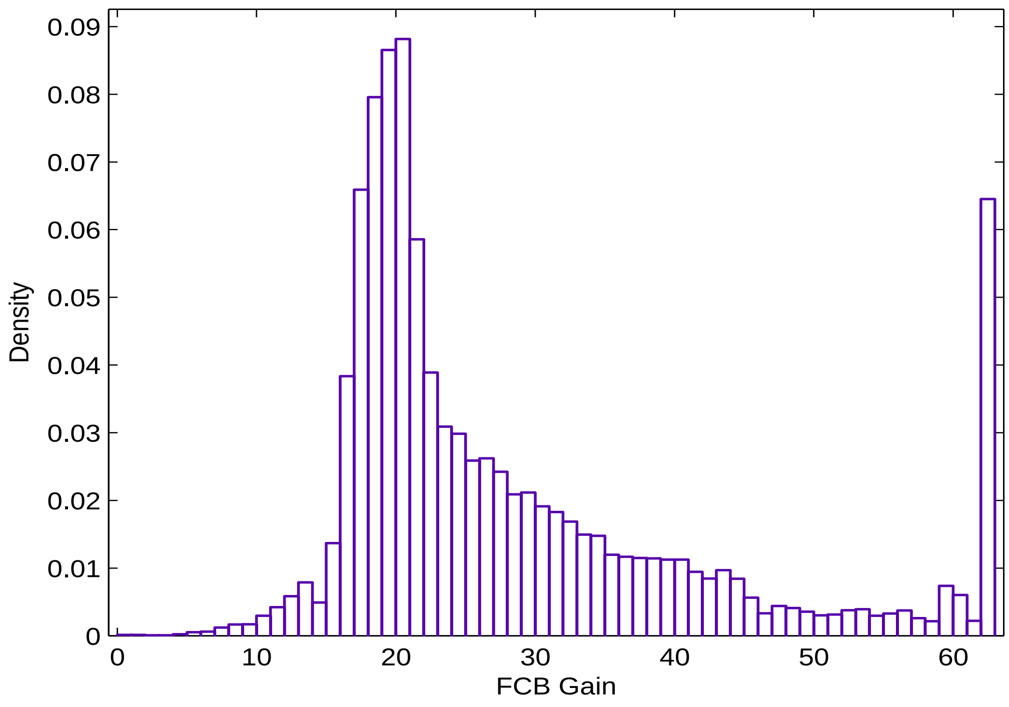

Figure 4.8: Probability Density Function of Fixed Code Book Gain for 4 sub frames

decoding part which is based on these probabilities will rightly yield different results for the four different frames.

4.3.2 Decoding Phase

Once the models for each component of the speech frame, in the form of bi-gram probabilities, are available, then these can be used to find the individual components pi and qi of two different speech files given xi such that ; where l represents the number of speech frames to be decoded. Considering the decoding of the first component of the CELP frame in our case, which is the 1st LSF index representing LSF1 to LSF3 sub vector, the possible number of states at each stage is 10242. It means that we have to pair each alphabet with every other alphabet (and off course with itself) to cater for all possibilities, but fortunately since we know x at each stage of the resultant automaton formed by the intersection of two different automata, we only select those pairs which satisfies the relation . Hence the total number of pairs of our interest at each stage again reduce to 1024. i.e. we will only consider the pairs of the form $(0, 0 \oplus x), (1, 1 \oplus x), \cdots (1023, 1023 \oplus x)$. This means that there are 1024 possibilities for each LSF subvector. Now for 1000 frames of speech signal, we will have 1000 values as 1st LSF sub vector (each of 10 bits). In the keystream reuse situation, we will have 1000 frames each of two different speech signals encrypted with the same key. If we take the first ten bits of each frame from all the 1000 frames of each speech signal and bitwise XOR them, then we get another 1000 values each of ten bits (lets say x=x1, x2, x3 ,, x1000;) which represent 1st LSF subvector of speech 1(p) XORed with 1st LSF subvector of speech 2 (q) i.e. x=p q. Now the job of the cryptanalyst is to first list down all possible pairs (p, q) for 1000 frames such that p q=x in a trellis of 1000 steps and then find out the most probable path through the trellis using the a priori information of probability distributions of p and q. The trellis of our interest for 1000 frames of speech will look like something shown at figure 4.9.

In the trellis shown in figure 4.9 the number of stages are equal to the number of frames we want to decode. For one minute of speech we will have 3000 frames (if we have a sampling rate of 8000 samples per second and the frame length is 20 millisecond). At each stage of the frame we will have 1024 nodes and between any two stages we will have 1024 x 1023 arcs between the adjacent nodes of the two stages. The weights of the arcs $a_{i(j,k)}$ as shown in figure 4.9 can be calculated for each stage as:

$$a_{i(j,k)} = Pr_1(k|j) \times Pr_2((k \oplus x_{i+1})|(j \oplus x_i)) \tag{4.1}$$

here Pr_1 and Pr_2 are the probability distributions pre-calculated and stored during the training phase. For the decoding part of our algorithm we used Viterbi Algorithm which guarantees to find the optimal path through a trellis. In order to apply Viterbi algorithm one needs to have a transition matrix which lists all the transitional probabilities between all possible states of the trellis. In

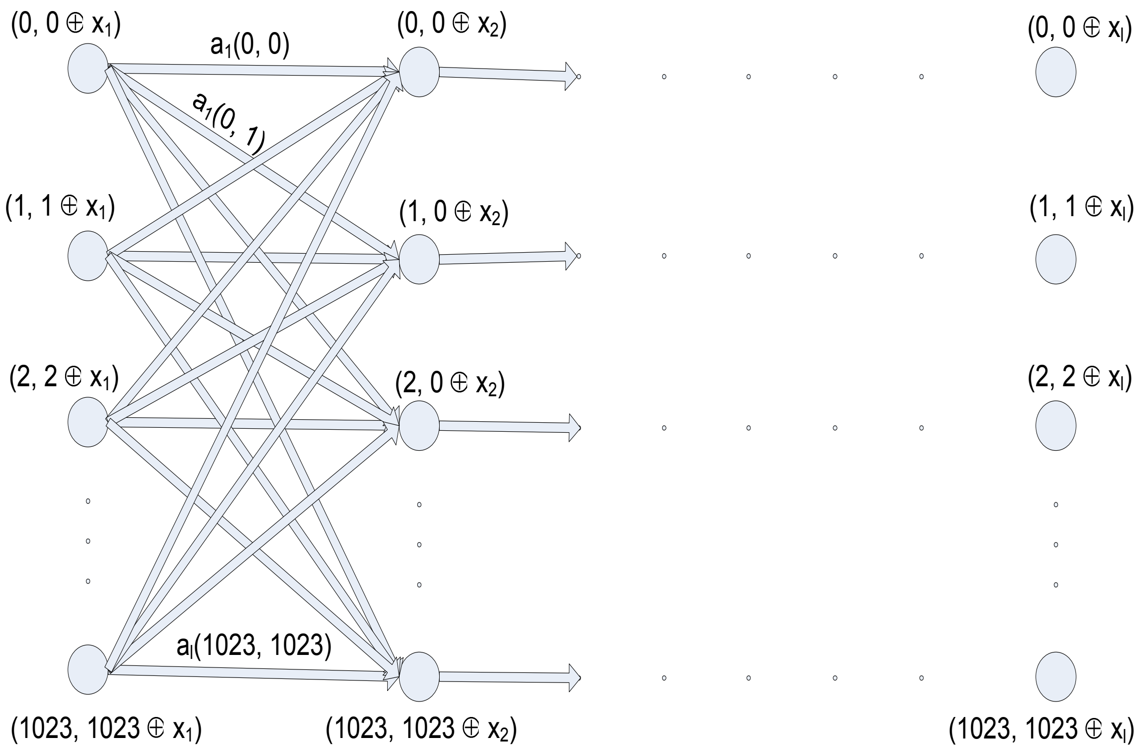

Figure 4.9: Trellis Diagram of the First LSF Index for 1000 frames of speech

order to cater for the XORed situation of the first LSF index in which 1024 different values are paired with another 1024 values, we need to define an array of 10242 x 10242 dimensions. Pre calculating such a huge matrix and storing it is not advisable and requires a lot of memory. We calculated the individual transition matrix of 1024x1024 dimensions while the calculations of transitions at every stage are done on the fly at each stage of the trellis during the course of the Viterbi search for optimal path. Since all 47 components of our CELP frame are modeled separately in our implementation, hence these are to be separately decoded as well. For each component a separate trellis of the form shown in figure 4.9 has to be constructed.

4.3.3 Experimental Results

As discussed earlier, for the training phase of our bi-gram models we used the SWITCHBOARD corpus. For the test purpose, we selected other ten file not included in the training data while

Table 4.2: Decoding Accuracy of different parameters of the CELP Frame

Parameters	Decoding Accuracy (%)					
	Test Files selected from Training Data			Test Files not from Training Data		
	No Smoothing	Additive	WB	No Smoothing	Additive	WB
1st LSF subvector	53.2	61.4	68.8	5.2	56.7	62.4
2nd LSF subvector	49.8	56.3	62.3	7.5	49.5	57.1
3rd LSF sub vector	51.3	52.7	58.1	8.5	47.6	51.7
Pitch Delay	45.6	60.1	56.2	2.3	51.2	49.8
Pitch Gain	78.3	81.4	89.5	26.1	67.5	67.9
FCB Pulse Position	61.2	60.2	59.9	9.3	52.7	51.3
FCB Pulse Sign	3.6	3.8	3.1	4.1	0.5	2.3
FCB Gain	59.3	61.0	58.6	48.9	49.2	47.7

we also tested our decoding algorithm initially on ten files selected from the training data as an initial step. These files were bitwise XORed to simulate the keystream reuse situation and then decoded. Table II shows the decoding accuracy results of p and q deduced from x ( ). Since all the components are modeled separately and hence they are to be decoded separately. As evident from the table the accuracy of the pulse signs are quite poor as compared to the rest of the parameters, the reason for this is the uniform distribution of 0 (indicating negative pulses) and 1 (indicating positive pulses). The accuracy for the test data selected from the training data shows good accuracy results even without smoothing but the results for the test data not selected from the training data is very low for models trained without smoothing, hence for accurate results in practical scenarios smoothing would be mandatory.

Parameters Decoding Accuracy (Test Files selected from Training Data Test Files not from training Data No Smoothing Additive WB No Smoothing Additive WB 1st LSF subvector 53.2 61.4 68.8 5.2 56.7 62.4 2nd LSF subvector 49.8 56.3 62.3 7.5 49.5 57.1 3rd LSF sub vector 51.3 52.7 58.1 8.5 47.6 51.7 Pitch Delay 45.6 60.1 56.2 2.3 51.2 49.8 Pitch Gain 78.3 81.4 89.5 26.1 67.5 67.9 FCB Pulse Position 61.2 60.2 59.9 9.3 52.7 51.3 FCB Pulse Sign 3.6 3.8 3.1 4.1 0.5 2.3 FCB Gain 59.3 61.0 58.6 48.9 49.2 47.7 Table II: Decoding Accuracy of different parameters of the CELP Frame.

4.3.4 Complexity Analysis

The complexity analysis of our algorithm needs to be done from the pre-computation as well as the decoding point of view. Since the pre computation or training part has to be done once for a particular type of encoding and language, and the same models can be used again and again, hence we can afford to have large processing and memory complexities. Efforts will be required to reduce the time during the decoding phase of our attack. Using dynamic programming it is possible in $O(l)$ time to find the most probable path in a trellis comprising of l number of frames for one particular component. We can achieve this by using single source shortest path algorithm to find the shortest path to any state, taking the length of each edge as the negative of the algorithm of probabilities, so that minimizing the sum of lengths will be equivalent to maximizing the product of probabilities . The main issue here is the size of the automaton formed by the cross product of two finite state automata which will have N nodes and E edges given by the following equations:

$$N = l.2^{y(z-1)} \tag{4.2}$$

and

$$E = l.2^{yz} \tag{4.3}$$

where l is the number of frames to be decoded, y is the number of bits representing the component of the CELP Vector and z is the order of the n-gram model which represents that component of CELP vector. For example for the first LSF index in our implementation as shown in figure 4.9, $l = 1000$, $y = 10$, and $z = 2$ as we are using the bi-gram models for representation. As we have discussed earlier that the accuracy of our results can be enhanced by using n-gram models of higher order but it can easily be seen from Equation x and x that by increasing the order of the n-gram models we exponentially increase the size of the automaton.

4.4 Summary

In this paper, we presented the first ever implementation of plaintext XORs of the key reuse problem in case of speech encoded through the modern encoding techniques used for low bit rate communication. It has been shown that the standard techniques of language modeling and dynamic programming algorithm can be adapted to be used for cryptanalysis of keystream reuse in case of stream ciphered digitized speech. The experimental results though preliminary, yet have shown encouraging breakthroughs. Contrary to cryptanalysis in such situation for text based data, which is mainly based on heuristics, a more mathematical approach has been adopted. We have presented our technique for the CELP coded speech, but the same techniques may be adapted for use with other modern speech digitization and compression mechanisms. Since the Fixed Code Book Pulse signs have uniform distributions, hence our decoding algorithm could not produce satisfactory results, some different techniques may be looked into for decoding of the pulse signs as a future work. The option of joint distributions may be looked into.

Chapter 5

NETWORK SPEECH RECOGNITION APPROACH TO AUTOMATED CRYPTANALYSIS OF TWO TIME PADS

5.1 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. . If the keystream k is random and non-repeating then the stream cipher becomes provably unbreakable [36] and the cipher is called the one time pad. The security of a stream cipher rests on never reusing the keystream. If two different plaintexts p1 and p2 are encrypted with the same keystream k then their results and can be XORed to neutralize the effect of the keystream k, thereby obtaining . The key reuse problem in stream ciphers and its exploitation in different scenarios for the text based data have been studied since long. It has recently been mentioned in the literature as the two time pad problem [37]. The vulnerability of keystream reuse exists with many practical systems which are still in use. The practical systems which are vulnerable to such type of attacks include Microsoft Office [37, 28], 802.11 WEP [39], WinZip [40], PPTP [41]. The endorsement of AES counter mode by NIST [**?**] and IETF [44] has effectively turned a block encryption algorithm into stream cipher. Moreover, 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 in state of the art Giga-bit rate systems/links [44, 43]. Due to this compelling need of AES counter mode, the possibility of reusing keystreams has further increased.

The general technique for exploiting the vulnerability of keystream reuse situation for speech signals has recently been introduced by us in [86]. The referred technique has shown promising preliminary results and has paved the way for further research in this area. This paper is an endeavor in that direction to elaborate its application to practical secure speech communication systems. Since normal speech coders are only designed for speech communication and not recognition, hence speech signals transmitted from a highly variable acoustic environment after passing through a noisy wireless channel significantly degrades the performance of speech recognition even in the un-encrypted situation. The addition of some more distortion and distraction in the form of XORing the phonemes will worsen the situation. The past decade has seen a tremendous increase in the use of mobile and hand-held wireless devices. The user interface has also seen significant innovations but the use of hand-held devices has still been restricted and the main hindrances have been their miniature size and the mobility of the user. Speech recognition solves these limitations very efficiently by alleviating the mobile user from typing on tiny keyboards and pointing of stylus in an uncomfortable and error prone environment. Since all mobile devices use a communication link hence the automatic speech recognition (ASR) systems for mobile environment have been classified on the basis of location of the front-end (acoustic features extraction) and back-end (Viterbi search) processing. Three different architectures are in use with each having pros and cons in different situations. These are: client based (embedded) speech recognition, server based (network) speech recognition (NSR) and client-server based (distributed) speech recognition (DSR) [87]. In this paper we discuss the applicability of the NSR techniques in mobile environment to the automated cryptanalysis of the two time pads problem in stream ciphered digitized speech.

5.2 Proposed Approach

In an embedded ASR system the entire process of speech recognition is based on the terminal or mobile device which is generally applicable to relatively powerful devices such as PDAs. In distributed speech recognition (DSR) systems the acoustic front end processing is carried out at the client side, whereas the viterbi search is carried out at the server side. In network speech recognition (NSR), speech is transmitted over a communication channel and the recognition is performed on the remote server. The purpose of the NSR is to carry out speech recognition in case of thin clients by shifting both the acoustic feature extraction and the back end viterbi search algorithm to the fat server side. NSR provides access to recognizers based on different grammars and even different languages since all the computations are carried out by the remote fat server. Another advantage of NSR is that the recognition vocabulary might be secret and may not be appropriate to be kept at the client terminal. In spite of these advantages, NSR is the most unfavorite ASR architectures in mobile environment [87]. The reason for this is the performance degradation of the recognizer due to the low bit rate Codecs not particularly designed for ASR and further deteriorating the situation by transmission errors and background noise. But NSR is the only technique, out of the ASR architectures for mobile environment, which is directly applicable in our situation. Hence the otherwise phasing out server based ASR architecture (NSR) in mobile environment has to be revisited and revitalized by the telecommunication security professionals but from a very different perspective of speech recognition. The distortion introduced in speech during source coding can be catered for to a certain extent by training the ASR models with the corrupted speech. A better approach would be to carry out the recognition process on the basis of the features extracted from the parametric representation of the encoded speech. There are two main architectures for automatic speech recognition in mobile environment where the complete speech engine lies on the server. The division is based on the acoustic features extraction either from the decoded speech at the server or from the speech codec parameters without the use of speech decoder at the server. We will discuss both these architectures from the point of view of automated cryptanalysis of stream ciphered digitized speech in a keystream reuse situation.

5.2.1 ASR Feature Extraction After Speech Decoding

Figure 5.1 shows server based ASR architecture in which the acoustic front end processing is carried out after the decoding of the speech signal received from the data transmission link. This architecture corresponds to the actual architectures presented in [87] to [88] but with the addition of encryption. In this case the ASR features are extracted from the XORed speech signals being decoded by the speech decoder. In the plain speech domain, this type of arrangement shows a degraded performance due to the presence of data transmission errors and background noise. The best approach in this regards would be to train the recognizer under similar codec and noise conditions, hence the word error rate can be minimized by simulating the actual scenarios from coding and noise point of view but the combination of these conditions result in a large number of possible recognition scenarios. In this case we create our hidden Markov models on the same lines of [86]. However, the training of the HMMs is carried out with acoustic features extracted from the noisy decoded XORed speech signals.

5.2.2 ASR Features Extraction from Speech Codec Parameters

The distortion produced by the coders and the channel can be somewhat compensated by training the recognizer in a similar situation but this leads to very large number of models to be trained for the diversified combinations of coders and noisy conditions. A better approach is to extract the ASR features directly from the codec parameters [89, 90**?**]. Fig. 2 displays the architecture

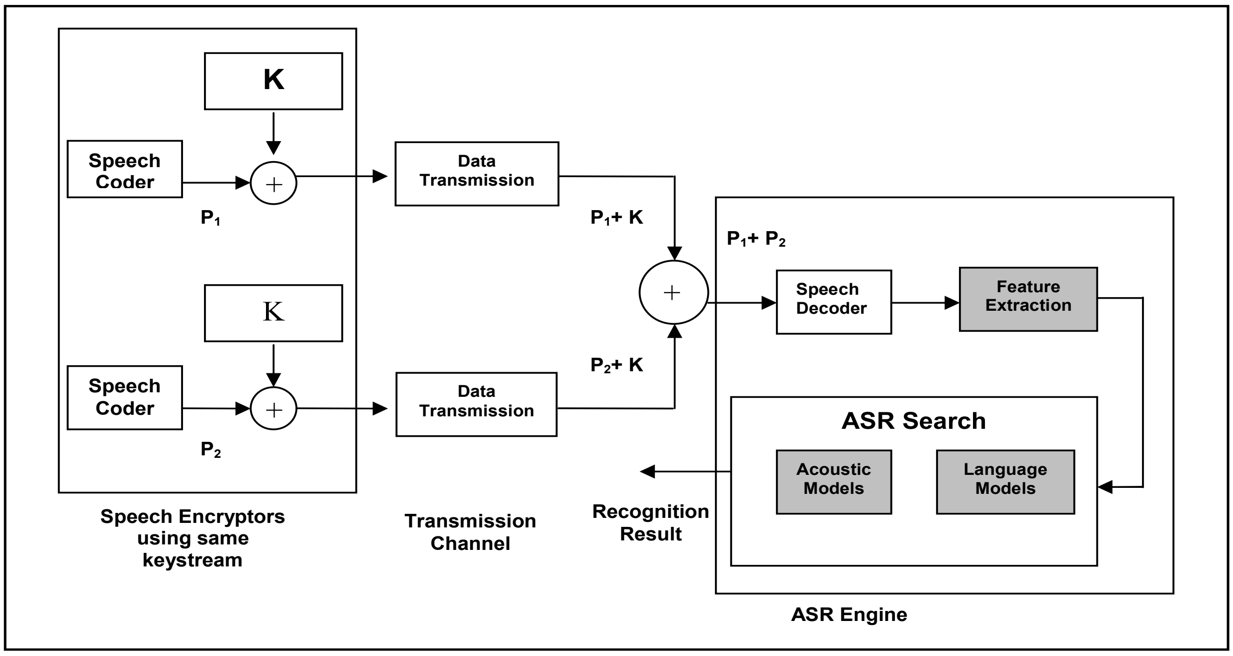

Figure 5.1: ASR features being extracted from the decoded XORed speech

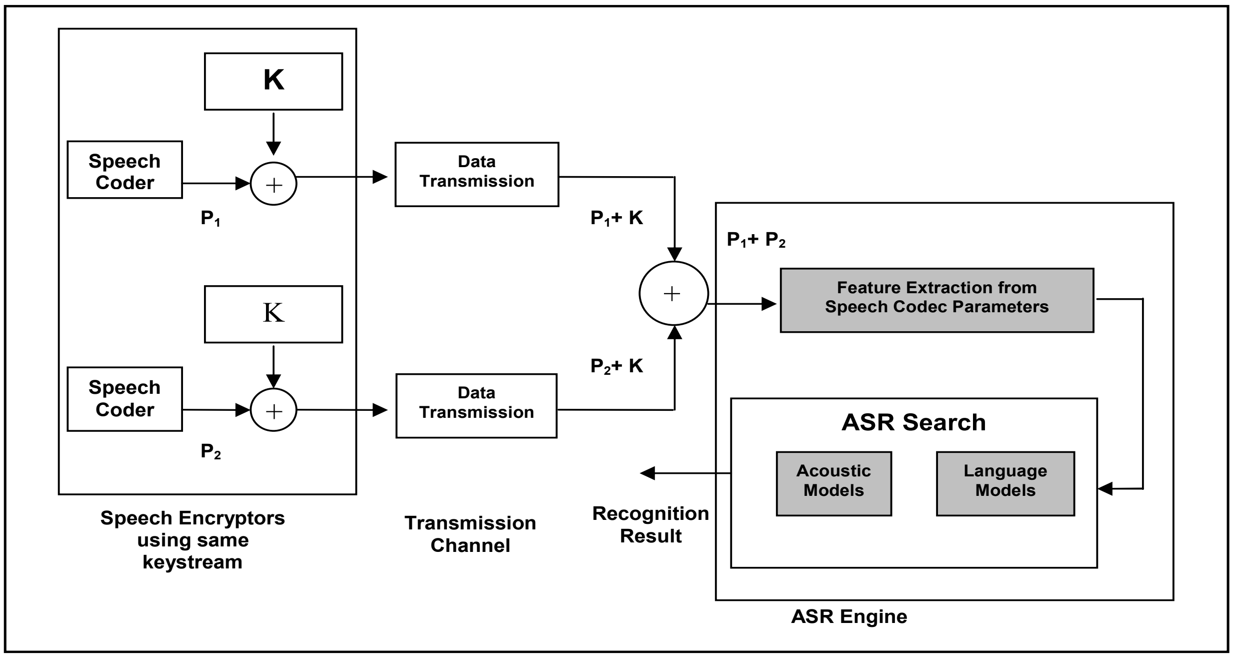

Figure 5.2: ASR Features being extracted from Codec Bit streams

which is used for speech recognition based on this concept but in keystream reuse situation. It is worth mentioning here that speech coding is based on the analysis of short term and long term information in the speech signal whereas speech recognizers employ only short term information as ASR features [89]. Different optimization techniques for different standard codecs are presented in the literature e.g. ITU-T G.723.1[89], ETSI GSM 06.10, CELP 3.3 [90], FS1015/1016 [91]. All these techniques can be applied in the keystream reuse situation also.

5.3 Simulation Results

For simulations we used HTK [75] which is an open source toolkit based on C language, available for building hidden Markov models (HMMs) and is primarily designed for speech recognizers. The simulation is comprised of two phases: the pre-computation phase which is basically the training of the recognizer whereas, the decoding phase corresponds to the Viterbi beam search of the most optimal paths in the recognition network. The accuracy of the attack is greatly dependent on how well the selected HMMs are trained and how accurately the simulated environment matches the actual situation. To simulate the keystream reuse situation, we selected 256 files from the SwitchBoard corpus [77]. The reason for this selection was to simulate the environment of eavesdropped conversational speech. We first XORed different pairs of the selected files to simulate the plaintext XOR situation. Fig. 3 shows the spectrogram and waveform obtained after XOR of two speech files selected from our corpus along with their transcription in the form of phoneme pairs. In the training part of our simulation, we first selected our HMMs, the basic of architecture of which is shown in Fig. 4. We then trained these HMMs on the transcribed SwitchBoard files obtained after XORing. A total of 588 HMMs were trained with each HMM corresponding to one phoneme pair. For testing purpose we selected ten different speech files from the Switchboard Corpus which were not included in the training data. An initial test was

Table 5.1: Recognition Accuracies

Feature Extraction Procedure	Recognition Accuracy (%)	
	Test Files selected from training data	Arbitrary Test Files
From Speech Files without transmission	81.53	76.41
From Decoded Speech Files after transmission	71.81	61.26
From Codec Bit streams after transmission	69.32	58.21

also performed on ten speech files selected from the training data. The experimental results of the recognition of the XORed phonemes are depicted in Table 5.3. The front end processing on the codec bit stream shows a relatively poor performance as compared to the front end processing on the decoded speech.

5.4 Summary

In this paper we presented how the keystream reuse problem can be exploited by an adversary in case of stream ciphered digitized speech in a secure speech transmission system. The text based plaintext XORs have been discussed in the literature for quite some time now and the techniques have matured quite well. The network speech recognition (NSR) approach, although, not favorite for ASR in mobile environment, has shown to be directly applicable in this case. All the optimization endeavors for different Codecs in the plain speech scenario are also directly applicable in the keystream reuse situation. The experimental results, though of preliminary nature, have showed promising results, thereby paving the way for further deliberations on the topic.

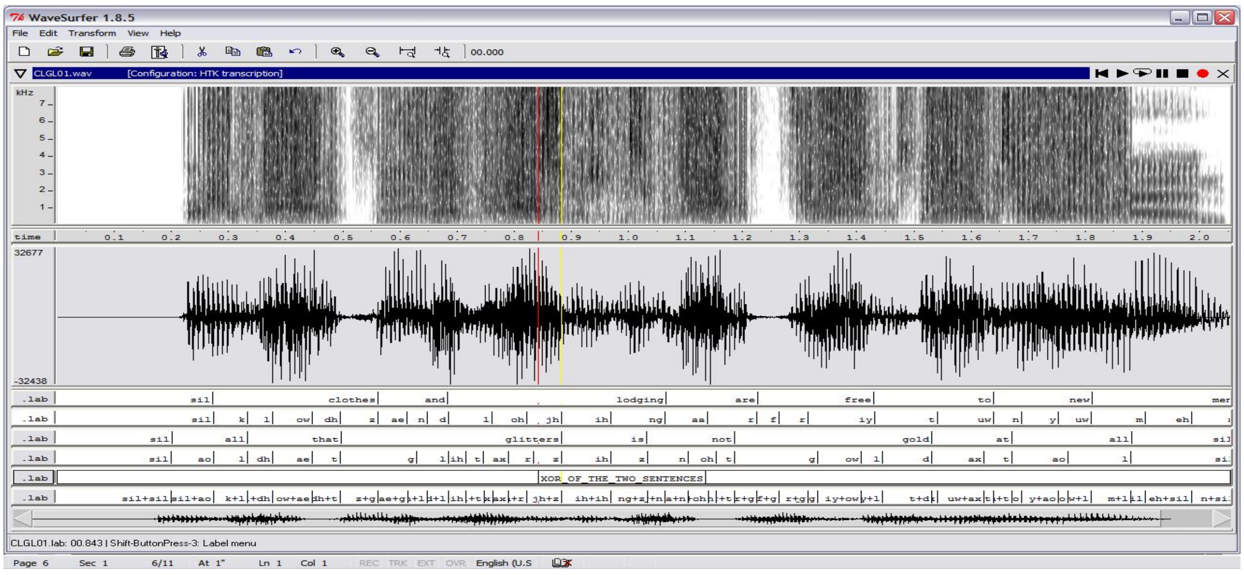
WaveSurfer 1.8.5
File Edit Transform View Help
00.000
CLGL01.wav [Configuration: HTK transcription]
kHz
time
32677
-32438
.lab
sil clothes and lodging are free to new mer
sil k l ow dh z ae n d l oh jh ih ng aa r f r iy t uw n y uw m eh
sil all that glitters is not gold at all sil
sil ao l dh ae t g l ih t ax r z ih z n oh t g ow l d ax t ao l si
XOR_OF_THE_TWO_SENTENCES
sil+sil sil+ao k+l +dh ow+ae dh+t z+g ae+g +l +l ih +t ax +r jh+z ih+ih ng+z +n +n oh +t +g +g r+g g iy+ow +l t+d uw+ax t +t o y+ao o w+l m+l l eh+sil n+si
CLGL01.lab: 00.843 | Shift-ButtonPress-3: Label menu
Page 6 Sec 1 6/11 At 1" Ln 1 Col 1 REC TRK EXT OVR English (U.S

Figure 5.3: Spectrogram and waveform along with transcription of XOR of the sentences

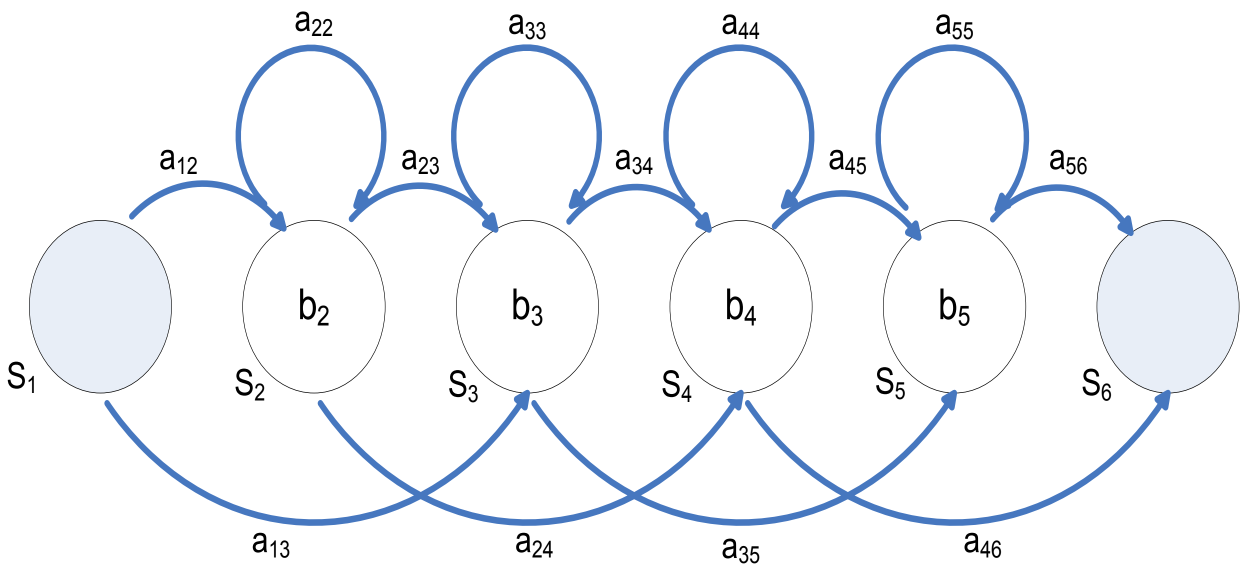

Figure 5.4: Basic Architecture of the Hidden Markov Model used in our experiments

Chapter 6

SPEAKER RECOGNITION FROM PARTIALLY ENCRYPTED SPEECH

6.1 Introduction

Various methods involving numerous measurement techniques and signals have been proposed and investigated for use in forensic identification and verification systems. Among the most popular measurements are finger prints, DNA and voice. While each one has pros and cons with respect to accuracy and deployment mechanism, there are two main factors that have recently introduced speech as a popular and upcoming source of forensic identification and verification [92]. First, speech is a natural signal to produce and is not considered threatening by users to provide, it may be collected without the subject being aware of [93]. Second, the telephone system (including voice over IP applications over the internet) provides a ubiquitous and well established network of sensors for obtaining and delivering speech signal. In many applications the available speech may be the only forensic evidence available especially in cases of threatening, teasing or blackmailing someone or also in case of linkages to heinous crimes like drug trafficking and terrorist activities. Moreover, as pointed out in [94], many forensic cases boil down to the question whether a set of recordings some of which are from the perpetrator and others from a single suspect, do or do not originate from the same speaker. In such cases the role of automatic speaker verification in forensic investigation becomes extremely useful.

Over the past few years, voice over IP has become an attractive form of telephony. Since VoIP calls may traverse through untrusted networks, the need for encrypting the packets has increased manifold [95]. Since the packet has to be encrypted and decrypted several times before it reaches its destination, low computational encryption techniques have become the need of the hour. Moreover, with the advancement in mobile multimedia applications, the demand for privacy and content protection with lean computational load and low power consumption is also on the rise. This demand has generated intense research in the design of low complexity encryption techniques [96][97]. One of such techniques for speech encryption is the low complexity perception based partial encryption technique presented in [78] applied on the mostly deployed speech codec recommended by the International Telecommunication Union vide ITU-T G.729 [98]. This encryption technique is applied on a fraction of the bit stream and lowers the computational load, thereby freeing resources for other tasks and/or extending battery life. These partial encryption techniques, may provide confidentiality as far as content protection is concerned but we show that it may not conceal (completely) the identity of the speaker and can still be used in forensic speaker verification with a fair degree of certainty. In this paper, we are studying the problem of speaker verification by focusing on two forensically significant scenarios: First, given two partially encrypted conversations, out of which, one is known to belong to a perpetrator and the other segment is to be determined if it also belong to the same speaker or not? [94]. Second, given a potential suspect allegedly associated with one or more pre-identified perpetrators, the investigator assumes to have access to a set of (perception based partially) encrypted conversations (tapped from a telephone line or cellular devices) of the perpetrator with the suspect. The problem is to analyze the available speech conversations and verify that these belong to the suspect i.e. the suspect has conversed with the criminal. We explore the possibility of verifying the speaker identity from the perceptually irrelevant portion of the speech signal which is not encrypted in partial encryption techniques with a view that it (in isolation) contains little or no significant information regarding the speech content or the speaker. The popular techniques of text independent speaker verification based on hidden Markov Modeling [99], Gaussian Mixture Modeling [100]and adapted Gaussian Mixture Modeling [101] are applied on the perceptually irrelevant portion of the speech signal. Although the verification accuracies are not comparable to those obtained from the perceptually relevant portion of the speech but at the same time these are not as inaccurate as obtained from random data as expected in case of encrypted speech. We show that even these perceptually irrelevant features of the speech signal convey sufficient information (in isolation) to reveal or confirm the identity of the speaker.

The rest of the paper is organized as follows: Section 7.2 presents the background information regarding speaker verification and the modeling techniques used in this paper. In Section 6.3, we present our proposed approach to apply the existing speaker verification techniques on the partially encrypted speech files. Section 6.4presents the experimental results obtained by comparing the different approaches of automatic speaker verification systems. The paper is concluded in Section 7.7 and directions for future research are also presented.

6.2 Related Work

In this section we present background information regarding speaker verification, hidden Markov and Gaussian Mixture models which we use for modeling the unencrypted portion of the speech.

6.2.1 Speaker Verification

Given an observation sequence in the form of a speech segment O, and a hypothesized speaker S, the task of speaker verification is to determine if O was spoken by S [102]. Mathematically, the task of speaker verification can be termed as a basic hypothesis test between

H_0: O is from the hypothesized speaker S and

H_1: O is not from the hypothesized speaker S.

The optimum test to decide between these two hypotheses is a likelihood ratio test given by

$$\frac{p\,(O/H_0)}{p\,(O/H_1)} \geq \theta,$$

accept H_0, otherwise reject H_0 (accept H_1) where $p(O/H_i), i = 0, 1$ is the probability density function for the hypothesis H_i evaluated for the observed speech segment O and θ is the decision threshold for accepting or rejecting H_0. The basic goal of the speaker verification system is to find techniques for finding the two likelihood functions $p(O/H_0)$ and $p(O/H_1)$. Various approaches have been in place for finding these two likely hood functions, hidden Markov and Gaussian Mixture models being the most common ones. Two types of errors can occur in the speaker verification system namely false rejection (rejecting a valid speaker) and false acceptance (accepting an invalid speaker), the probability of both being termed as P_{fr} and P_{fa}, respectively. Both the errors depend on the value of the threshold θ. It is therefore possible to represent the performance of the system by plotting P_{fa} versus P_{fr}, the curve generally known as the detection error trade offs (DET) curve. In order to judge the performance of the speaker verification systems different performance measuring means are in place, among which the Equal Error Rate (EER) and Minimum Detection Cost Function $(minDCF)$ are the popular ones. The EER corresponds to the point where $P_{fa} = P_{fr}$. The $minDCF$ punishes strictly the false acceptance rate and is defined as the minimum value of "$0.1 * falserejectionrate + 0.99 * falseacceptancerate$" [103].

6.2.2 Hidden Markov Models

A Markov model is a model in which the system to be modeled is assumed to be a Markov Process : a stochastic process in which the future state of the process depends only on the present states and not on any past state. A hidden Markov model (HMM) is a statistical model in which the system to be modeled is assumed to be a Markov process with hidden parameters and the challenge is to find the hidden parameters of the model from the observable parameters. An HMM is characterized by the following [104]:

(a) N, the number of states $\{S_1, \cdots, S_N\}$ in the model. Although the states are hidden in an HMM yet in every practical system some physical significance is attached to the states. Generally the states are connected in such a way that any state can be reached from any other state. We denote individual state at time t as q_t.

(b) M, the number of distinct observation symbols per state, generally known as the alphabet size. The observation symbols correspond to the physical output of the system being modeled. We denote the individual symbols as $V = \{v_1, \cdots, v_M\}$.

(c) The state transition probability distribution $A = a_{ij}$ where

$$a_{ij} = P\left[q_{t+1} = S_j / q_t = S_i\right] \ \ , \ \ 1 \leq i, j \leq N. \tag{6.1}$$

For the special case that any state can be reached from any other state $a_{ij} > 0$ for all i, j.

(d) The observation symbol probability distribution in state j, $B = b_j(k)$, where

$$b_j\left(k\right) = P\left[v_k \,\text{at}\; t / q_t = S_j\right] \ \ , \ \ 1 \leq j \leq N, 1 \leq k \leq M \tag{6.2}$$

(e) The initial distribution $\pi = \{\pi_i\}$ where

$$\pi_i = P\left[q_1 = S_i\right] \ , \ 1 \leq i \leq N \tag{6.3}$$

It is evident from the above discussion that an HMM can be completely described by two model parameters M and N and three probability measures A, B and π . For convenience we use the compact notation

$$\lambda = P\left(A, B, \pi\right) \tag{6.4}$$

to indicate the complete parameter set of the model.

6.2.3 Gaussian Mixture Models

A Gaussian mixture density is a weighted sum of M component densities and given by the equation [101]:

$$p\left(x/\lambda\right) = \Sigma_{i=1}^{M} p_i b_i \tag{6.5}$$

where x is a D-dimensional random vector,$b_i(x), i = 1, \ldots, M$, are the component densities and $p_i, i = 1, \ldots, M$ are the mixture weights in the Gaussian mixture model λ. Each component density is a D-variate Gaussian function of the form

$$b_i\left(x\right) = \frac{1}{\left(2\pi\right)^{\frac{D}{2}} \left|\Sigma_i\right|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(x-\mu_i\right)' \Sigma_i^{-1}(x-\mu_i)\right\} \tag{6.6}$$

with mean vector μ_i and covariance matrix Σ_i. The mixture weights satisfy the constraint $\sum_{i=1}^{M} p_i = 1$. The complete Gaussian mixture density is parameterized by the mean vectors, covariance matrices and mixture weights from all component densities. These parameters are collectively represented by the notation

$$\lambda = \left\{p_i, \mu_i, \Sigma_i\right\}, i = 1, \ldots, M \tag{6.7}$$

The GMMs can have different forms of covariance matrices. These can be full or diagonal matrices. We used GMM with diagonal covariance matrix in this paper.

6.3 Speaker Verification from Perceptually Irrelevant Speech

Figure 6.1 shows the block diagram of our speaker verification system. In a conventional speaker verification system the front end processing is to extract speaker dependent information usually Mel frequency Cepstral coefficients [105] from the speech signal which can be used for classifying whether the input speech segment belongs to the hypothesized speaker or otherwise. Techniques to minimize the confounding effects from these features such as linear filtering [105] or noise may also be employed in the front end processing. In our case we are confronted with two main issues in this domain. First, the input speech signal is perception based partially encrypted speech in which the perceptually relevant portion is just noise. We are left only with the perceptually irrelevant portion of the speech. Secondly, we are confronted with a specific encoding and compression mechanism of ITU-T G.729, contrary to the speech waveform or Mel Frequency Cepstral features for which speech and speaker recognition applications have proven better performance [105]. In our speaker verification system the feature extraction corresponds to extracting the perceptually irrelevant (unencrypted) portion from the CELP coded frames to simulate the partial encryption scenario. Figure 6.2 (a) shows the partially encrypted frame with the high encryption rate of [97] in which only 6 out of the 15 parameters per frame are left un-encrypted while the rest are either completely or partially encrypted. The feature extraction in our case boils down to extracting the feature frame as shown in Figure 6.2 (a). The description and bit allocations for the features are shown in Table 6.1 with the encrypted bits shown in grey.

By analyzing the distribution of the unencrypted parameters for different speakers, it was found that these distributions display a non uniform structure. Figure 6.3 shows the histograms and

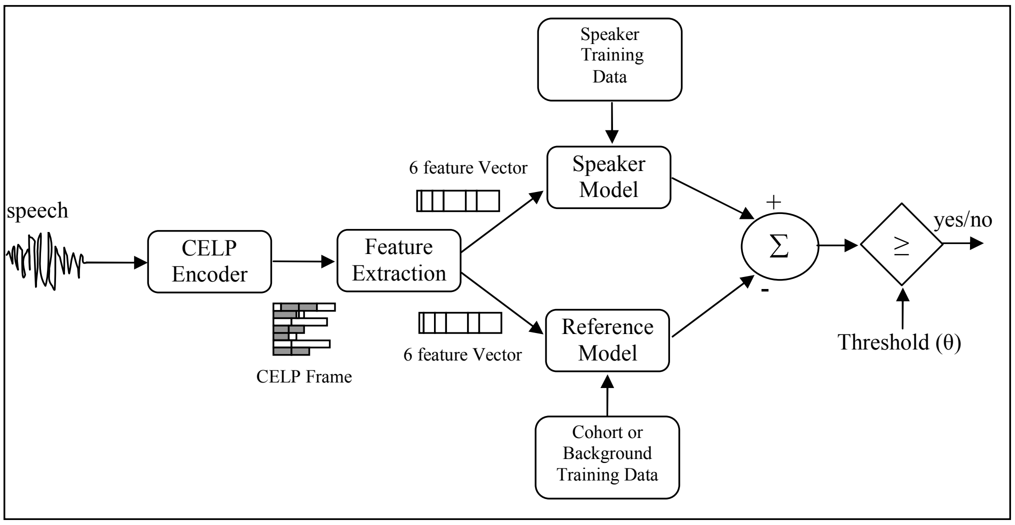

Figure 6.1: Block Diagram of the proposed speaker verification system

Frame/subframe	Symbol	Description	Bits
Per Frame	L0	Switched LSP quantizer MA predictor index	1
	L1	First code book index	7
	L2	Second LSP Code book index (lower vector)	5
	L3	Second LSP Code book index (higher vector)	5
1st Sub-frame	P1	Pitch Delay	8
	P0	Parity bit for pitch delay	1
	S1	Fixed Codebook Pulses signs	4
	C1	Fixed Codebook Pulse Position	13
	GA1	Gain Codebook (1st stage)	3
	GB1	Gain Codebook (2nd stage)	4
2nd Sub-frame	P2	Pitch Delay	5
	S2	Fixed Codebook Pulses signs	4
	C2	Fixed Codebook Pulse Position	13
	GA2	Gain Codebook (1st stage)	3
	GB2	Gain Codebook (2nd stage)	4

Table 6.1: Bit allocation for various parameters in one frame (with 2 subframes) of ITU-T G.729. (The encrypted parameters are marked grey)

probability density functions of the parameter L3 (which corresponds to the second part of the line spectral frequency in the CELP frame) for two different speakers in our test corpus. The non uniform distribution of the parameters encourages us to statistically model these parameters. Out of the various modeling techniques available, we tried to model these features as hidden Markov and Gaussian Mixtures. The adapted Gaussian mixture modeling has shown better performance in terms of equal error rates and minimum cost detection function.

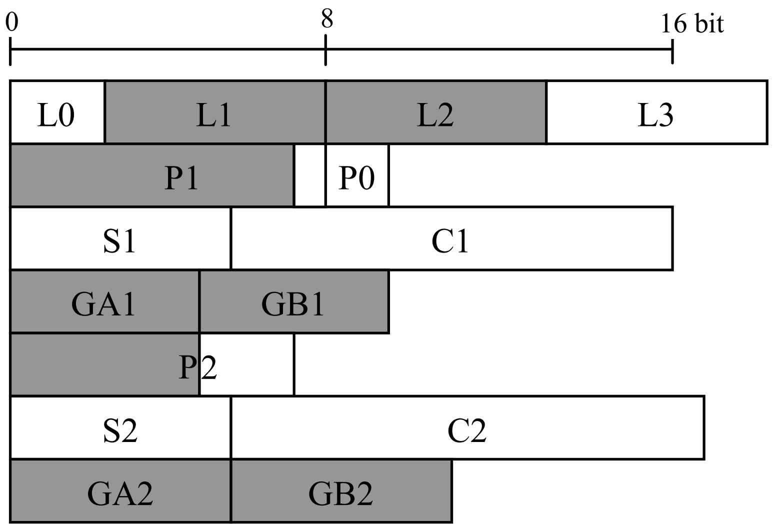

Figure 6.2: (a) Partial Encryption scheme of ITU-T G.729 single frame where the encrypted parameters are shown in grey. (b) shows the unencrypted frame comprising of six features which we used during our modeling.

6.3.1 Hidden Markov Modeling Approach

The hidden Markov modeling of speech parameters have been successfully used in various applications of speech processing particularly for speech recognition. For speech recognition applications HMMs have proved to be the most successful [104]. The HMMs not only very well model the underlying speech sounds but also the temporal sequencing of these sounds which is significant in case of text dependent speaker recognition but in case of text independent speaker recognition the sequencing of sounds found in the training data does not necessarily reflect the sequences of sounds in the test data and contains little speaker dependent information. In our case also, since we are dealing with text independent speaker verification, ergodic (fully connected) HMMs were used to model the sequence of vectors, each comprising of 6 features. In speaker verification the selection of cohort or world model plays an important role in the accuracy of the speech verification system [102]. The cohort is generally defined as the group of speakers whose models are considered to be close or more competitive with the hypothesized speaker. A different

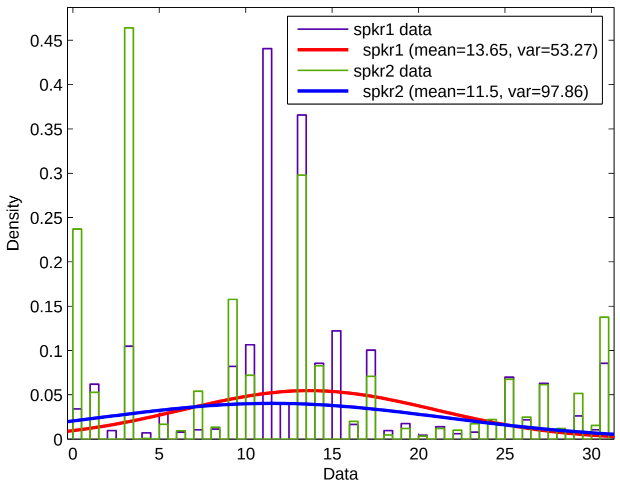

Figure 6.3: Histogram of a partial line spectral frequency component for a 50 second utterance by two different speakers.

cohort is thus assigned to every speaker and is determined during the training phase. Another approach is to estimate a speaker independent model generally termed as universal background model (UBM) in addition to the model for the hypothesized speaker and carryout classification of the input speech segment whether to belong to the hypothesized speaker or the UBM. We created two reference models one for female and the other one for the male speakers in our database. We used HMMs with different number of states and number of Gaussians per state for speaker verification.

6.3.2 Gaussian Mixture modeling Approach

The Guassian Mixture modeling approach for speaker verification for unencrypted speech has proven to be one of the most promising and accurate [100]. For the perceptually irrelevant speech in our case we tried to models the 6 dimensional vector per frame of the partially encrypted speech file as GMM for the hypothesized speaker with a speaker independent universal background model as the reference model. Various mixture densities starting from 8 upto 128 mixtures were used for modeling. The equal error has shown significant rise with the increase in the number of mixtures from 8 to 32 whereas above 32 the increase in accuracy declined and displayed insignificant improvement beyond 32 mixture densities. The mean EER in case of GMM could be brought to 27% and minDCF to 9%.

6.3.3 Adapted Guassian Mixture Modeling Approach

The adapted Gaussian mixture modeling approach has shown significant improvement in the context of using GMMs for speaker verification [101]. In this case a universal background model was trained using the perceptually irrelevant portion from speech files of the complete database. The speaker dependent GMMs were derived from this UBM through adaptation by using the perceptually irrelevant features from the speaker's training speech. In the adapted GMM case, the perceptually irrelevant features of the speech showed a significant improvement in equal error rates and minimum cost detection function from 8 to 32 mixture densities whereas the increase in accuracy was insignificant from mixture densities of 32 and higher.

6.4 Experimental Results

The experiments were performed on an 84 speaker AN4 database [106]. The AN4 database was recorded at Carnegie Mellon University from 24 female and 60 male speakers, each speaker having 13 speech files from 4 to 20 second duration in raw PCM format. Although the database is primarily collected for speech recognition in which 74 speakers data is marked for training while 10 speaker files are reserved for testing. We used the entire data of 84 speakers for speaker verification, bifurcating the 7 files per speaker for training and 6 files per speaker for testing purposes. In order to simulate the perception based partial encryption of CELP coded speech, the speech files were first encoded using an open source implementation of ITU-T G.729 encoder by VoiceAge [107]. The six features per frame which are considered as perceptually irrelevant were separated from each frame and speaker verification experiments were conducted using these features as input to our trainer and classifier. For this purpose we wrote a file in Matlab which extracts the feature from the encoded file and appends header to it recognizable by HTK. The six feature vector per frame for each speaker was used for training and testing the models. The experiments can be divided into two main categories as HMM and simple/adapted GMM tests. The typical DET curves for HMM, GMM and adapted GMM for one of the test speakers are shown in Figure 6.4. Conventionally, the DET curve is drawn on a deviate scale, since in our case the EER is above 20% in the best case, we used a normal scale.

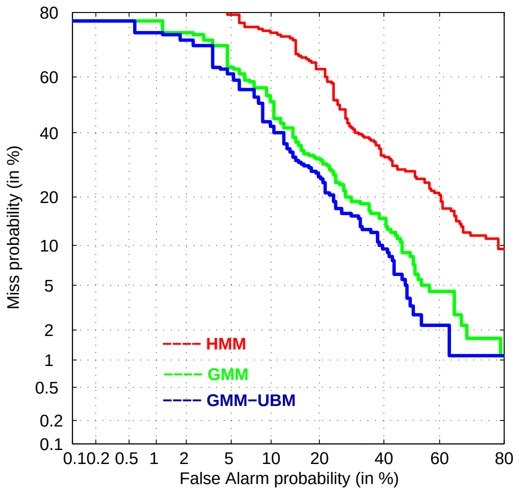

Figure 6.4: Typical DET curves for HMM, GMM and adapted GMM for one of the speakers in our test data.

6.4.1 HMM Tests

For the HMM modeling we used HTK [?] which is an open source toolkit based on C language available for building hidden Markov models particularly for speech recognizers. Seven of the training speech files per speaker in the AN4 database were used for training and six were used for testing. The tests were conducted by modeling the feature vectors as 3, 5 and 8 state ergodic (fully connected) HMMs. It is now well established that the models parameters are very sensitive to initial estimates [99]. An adequate choice for the initial estimates, which we also adopted, is to start with uniform distribution based on all the training files and then iteratively converge using the Baum Welch Algorithm [**?**]. The number of Gaussian mixtures per state in the HMM also plays a vital role and the optimum number varies in different situations. We performed the experiment on 1, 4 and 8 Gaussian mixtures per state. Figure 6.8 shows the Equal Error Rate as it varies with the number of states and the Gaussian mixtures per state in the HMM. The 3 state model showed comparatively better accuracies with less number of Gaussians per state. By increasing the number of states with less number of Gaussians per state reduced the verification accuracies. The 8 state only showed better accuracies with increased number of Gaussians per state. The reason for this could be the difference of temporal sequencing between the training and test data. The variation of minimum detection cost function with number of states and Gaussian mixture per state are depicted in Figure 6.6.

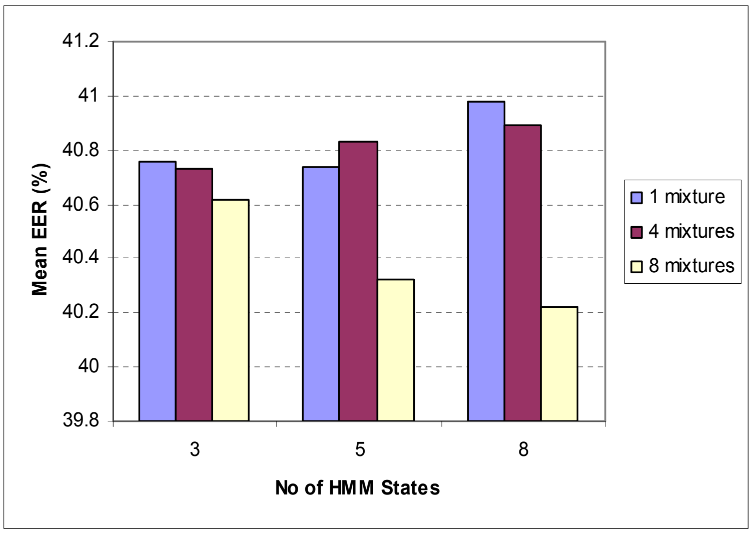

Figure 6.5: Variation of mean EER with number of HMM states and number of Gaussian mixtures per state.

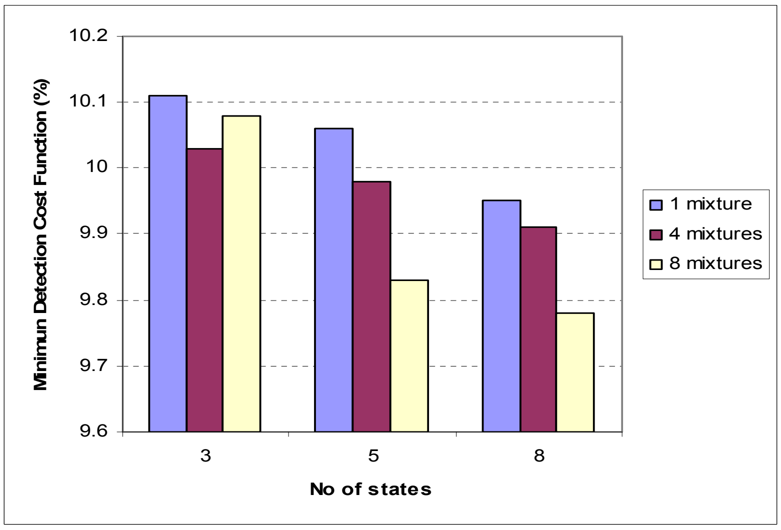

Figure 6.6: Minimum Cost detection function variation with HMM states and number of Gaussian per state.

6.4.2 GMM and Adapted GMM Tests

For the Gaussian mixture modeling we tricked the HTK by using single state HMM for each user thereby ignoring the temporal sequencing of the sounds in the training and test data. The tests were conducted by modeling the 6 feature vectors of each speakers as GMMs with 8 to 256 mixture densities. As expected, the GMMs displayed better verification accuracies in terms of EER and minDCF as compared to the HMMs. Since the GMM ignore the temporal sequencing of sounds in the data and model the distribution of different features as mixtures of Gaussians, thereby increasing the number of Gaussian densities per model increased the verification accuracies. In case of adapted GMMs we used a universal background model trained by the perceptually irrelevant portion of the speech segments from all the speakers and then adaptation of that UBM was carried out for each speaker. Figure 6.7 shows the mean equal error rate as a function of the number of Gaussian mixtures for the GMM and adapted GMMs. The EER improves with increasing the number of Gaussians per model upto 64 Gaussians but do not show any significant improvement beyond 64 mixtures.

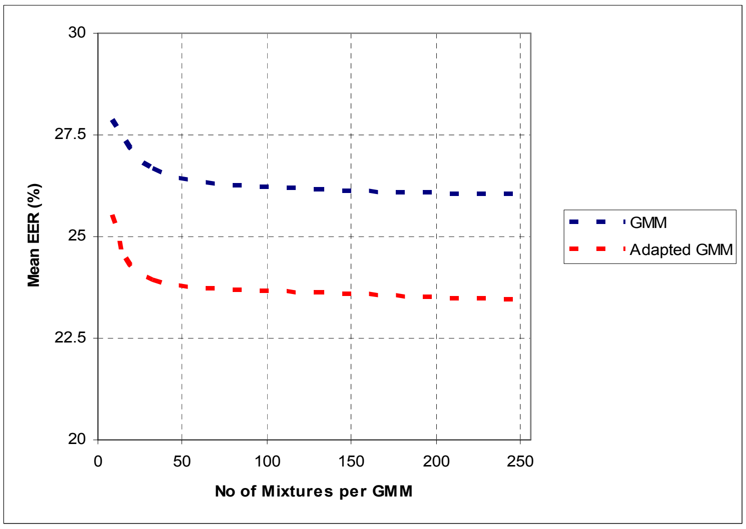

Figure 6.7: Variation of mean EER with number of Gaussian mixtures per model

The minimum cost detection function is depicted in Figure 6.8. With the GMM, the minimum cost detection function does not show any improvement with the increase in number of mixtures, whereas with the adapted GMMs, the minDCF improves from 8 to 56 mixtures per model and then shows no improvement beyond 56 mixtures.

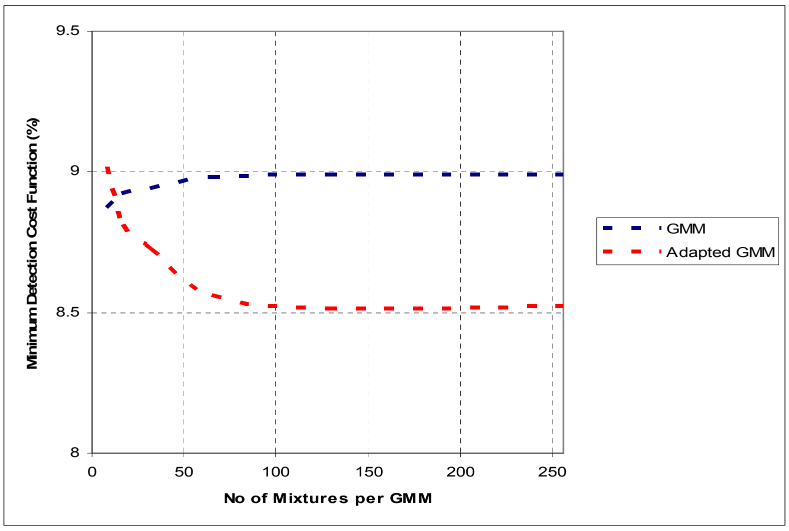

Figure 6.8: Variation of minimum detection cost function with number of Gaussian mixtures per model.

The variation of EER and minimum detection cost function with the model order (number of

Gaussian mixtures per model) is shown in Figure 6.9. The models orders from 8 to 256 are marked in the figure for both the modeling approaches. The adapted GMMs with 256 models display the minimum value for EER as 23.4% and minDCF as 8.4%.

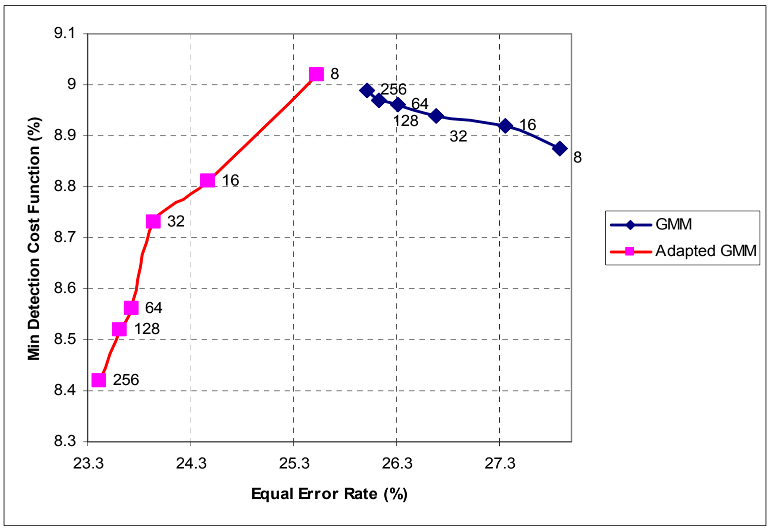

Figure 6.9: Verification accuracies of GMM and adapted GMM in terms of EER and minDCF with increase in model order (from 8 to 16).

6.5 Conclusion

Speaker verification from features of speech which are generally considered irrelevant from perception point view is presented. It has been shown that perception based partial encryption techniques may conceal the speech content but not the speaker's identity completely. The statistical modeling techniques such as HMMs and GMMs have proven to be useful in case of speaker verification. The adapted Gaussian mixture modeling technique with universal background model has shown better verification accuracies as compared to HMMs and simple GMMs. Perception based partial encryption techniques may be used for content protection but where complete confidentiality of the speaker is also intended, usage of such encryption techniques is not recommended. Other modeling and classification techniques such as vector quantization, Support Vector Machines, artificial neural networks etc may be taken up as a future work.

Chapter 7

SPEAKER RECOGNITION FROM ENCRYPTED VOIP COMMUNICATION

7.1 Introduction

Recent statistics show shrinking market share for traditional public switched telephone networks (PSTNs). This decline of PSTN market share is a direct result from the substitution of voice platforms as fixed wire-line operators migrate customers to all-IP voice platforms and as consumers opt for mobile voice platforms, which will also become all-IP. Unlike the traditional telephony systems where calls are transmitted through dedicated networks, voice over IP (VoIP) calls are transmitted through the Internet, a mix of public and private networks, which presents a threat to the privacy and confidentiality of VoIP communications. In order to overcome this problem, VoIP traffic is usually encrypted prior to its dispatch over the Internet [108]. Encrypting VoIP traffic, on one hand, helps preserving the privacy and confidentiality of legitimate users, but on the other hand might be exploited in criminal activities. Scammers, terrorists and blackmailers may abuse the end-to-end encryption facility to conceal their true identity. To address the problem of anonymity and to identify or confirm the true speaker of a disputed anonymous speech, the area of speaker recognition has long been the focus of forensic investigation. Research in speaker recognition has a relatively long history starting from introducing the term *voiceprint* identification [109] in 1962 to the tremendous development in the field of automatic speaker recognition in the last decade [110] which is marked by the National Institute of Standards and Technology

(NIST) evaluation campaigns [111, 112, 113]. Famous cases include the 1974 investigation of a conversation of the former US president, Richard Nixon, and former chief of staff, Harry Haldeman, which was recorded in the executive office building in 1972 [114]. The authentication of the speech recordings of Osama Bin Laden and other terrorists [115] using modern automatic speaker recognition techniques has also been the last resort to provide some forensic evidence in these recent cases.

Automatic speaker recognition can be divided into two categories: *identification* and *verification*. In the former scenario, given a set of suspected speakers together with their recorded speech segments, the problem would be to determine the likelihood that a disputed encoded speech segment belongs to one of these suspects. In the second scenario, a forensic investigator is given a disputed speech along with a set of recordings of a potential perpetrator and is asked to analyze if both sets of speech segments originate from the same individual [116]. Both scenarios are addressed in this paper but from the perspective of encrypted VoIP communications. Existing speaker identification and verification techniques are employed for analyzing un-encrypted speech only. However, to the best of our knowledge, there is no such study available for *encrypted* speech.

Variable bit rate (VBR) encoding techniques, which result into variable length VoIP packets, have been introduced to preserve the network bandwidth. The encryption techniques currently in use in order to preserve privacy of the calling and called parties do not change the packet length [117]. Hence any exploitation mechanism based on the packet length information remains valid in the encrypted communication. In this paper, we propose speaker identification and verification techniques based on using packet-length information without even knowing the contents of encrypted VoIP conversations. We demonstrate that the packet-length information, being extracted from either the file headers (in case of multimedia container formats) or being physically monitored during a VoIP conversation, can be used to identify or verify the speaker. In particular, we use discrete hidden Markov models to model each speaker by the sequence of packet lengths produced from her conversation in a VoIP call. Tri-gram probabilities of the packet length sequences were also used to create Gaussian mixture models and decision trees based on these probability distributions for each speaker. Various statistical modelling and classification/regression techniques were also applied, out of which the Ensemble of Nested Dichotomies (ENDs) achieved more than 10 fold improvement over random guessing in identifying a speaker from a group of 20 suspects. In case of speaker verification, an equal error rate of 17% was obtained using Support Vector Machine (SVM) based regression techniques.

The significant contributions of our approach are:

1. We are the first, to the best of our knowledge, to apply speaker identification and verification to *encrypted* VoIP conversations.

2. The recently developed container formats which are used to store and carry multimedia information over the Internet are explored from the perspective of speaker recognition in case of encrypted communications.

3. Our experimental results indicate that different types of classification and regression techniques, that are usually used in data mining and machine learning applications, outperform Gaussian mixture models and hidden Markov models-the classifiers which perform very accurately in the conventional speaker recognition studies.

The rest of the paper is organized as follows. In Section 7.2, we discuss the related work in the area of speaker recognition as well as packet-length information exploitation in encrypted VoIP conversations. The basic idea behind our work is discussed in Section 7.3. The problem statement of our work is presented in Section 7.4 and the proposed approach in explained in Section 7.5. Section 7.6 presents the experimental evaluation and the paper is concluded in Section 7.7.

7.2 Related Work

In this section, we briefly summarize the state-of-the-art in the field of speaker recognition. in two directions, namely, recently proposed techniques for automatic speaker recognition. Second, whether speech segments, used for speaker recognition, are encrypted or un-encrypted. Although significant work has been done in the area of speaker recognition, throughout this section, we only focus on two pertinent approaches: Gaussian Mixture Model Universal Background Model (GMM-UBM) [100], and Mixed GMM-UBM and SVM technique [118]. These models are commonly used in text-independent speaker recognition problems especially in speaker verification or source confirmation disputes. The mixed GMM-UBM and SVM approach combines the modelling efficacy of Guassian Mixtures and the discriminative power of SVMs and has shown significant improvement in terms of identification accuracies. In case of speaker identification, the accuracy measurement is simple and can be termed as the ratio of the correctly identified speech segments to the total number of segments in a group of speakers. This accuracy measure is greatly dependent on the potential number of suspects; increasing the population size reduces the accuracy. Speaker verification, being a two class classification problem, can generate two types of errors, namely false rejection (rejecting a valid speaker) and false acceptance (accepting an invalid speaker). The probability of both are denoted as P_{fr} and P_{fa}, respectively. Both errors depend on the value of the threshold set for classification. It is, therefore, possible to represent the performance of the system by plotting P_{fa} versus P_{fr}, a curve that is generally known as the detection error trade offs (DET) curve. In order to judge the performance of the speaker verification systems, different performance measures are in place, among which the Equal Error Rate

(EER) and Minimum Detection Cost Function $(minDCF)$ are the popular ones. The EER corresponds to the point where $P_{fa} = P_{fr}$. The $minDCF$ punishes strictly the false acceptance rate and is defined as the minimum value of $0.1\times$*false rejection rate*$+0.99\times$*false acceptance rate* [**?**]. Another noticeable work in the field of speaker recognition is the National Institute of Standards and Technology (NIST) Speaker Recognition Evaluation (SRE) framework used to evaluate different text-independent speaker recognition techniques and models [113]. It started in 1996 and continues till date.

As far as speaker recognition from encrypted speech is concerned, there is no study available, at least as of now. However, Wright et. al [95][119] have studied the utilization of packet-length information in extracting some crucial information about the encrypted VoIP traffic. In particular, the authors were able to identify the spoken language of the transmitted encrypted media with an average accuracy of 66%. In the second case, partial speech contents were extracted using packet-length information. Both these techniques do not disclose the identity of the speaker beyond her language of communication.

7.3 Main Observation

The basic idea behind this work stems from observing the relationship between the speaker's identity and the length of the packet carrying her VoIP speech contents. In order to save bandwidth, especially in case where, on average, 63% of the time one of the two channels in a VoIP call is idle, variable bit rate (VBR) encoding has been introduced [120]. VBR techniques allow a codec to change its bit rate dynamically to adapt to the acoustics of the audio being encoded. Sounds like vowels and high-energy transients require a higher bit rate to achieve good quality, while fricatives such as the *s* and *f* sounds can be coded comparatively with fewer bits. For this reason, for a given bandwidth, VBR can achieve lower bit rates for the same quality. As demonstrated in [119], the bit rate used for encoding speech and the length of the packets carrying it are in perfect synchronization. The reason for this is the fixed frame length (10-30 milliseconds, typically 20 milliseconds) in case of speech compression and encoding mechanisms used in VoIP communications. For example, a VBR encoder operating at 13.6 kbps will produce a packet-length of 34 bytes, excluding the header, for speech sampled at 8000 samples per second.

In order to determine the correlation of the speaker's identity with packet-lengths of the encoded speech, we conducted some experiments on the AN4 Low Vocabulary Speech Recognition Database [121] which consists of identical phrases uttered by different speakers. Fig. 7.1 shows the histogram of the packet lengths with Gaussian curve fitting of the same phrase uttered by three different speakers randomly selected from the AN4 speech recognition corpus and encoded by the VBR encoder, Speex [122], which encodes this phrase with eight distinct bit rates resulting into eight different packet lengths. It is interesting to note that different speakers produce different distributions for the same eight packet sizes. Various experiments were conducted with similar speech contents but uttered by different speakers. Throughout our experiments, the distribution of frame length sequences produced by different speakers were visually distinguishable. This dependence of the frame length sequencing on the speaker's identity encouraged us to model each speaker with respect to frame length sequences.

In order to understand the effect of encryption on the frame lengths, we studied the practically in-vogue security mechanisms in VoIP communication. One of the techniques proposed to secure real time speech communication over the Internet is to tunnel VoIP over IPSec but this proposal has a serious limitation of inducing unacceptable delays on the real time traffic [123]. In order to address the issue of confidentiality and privacy in VoIP communications without compromising the quality of service, the real time transport protocol (RTP) for multimedia applications was

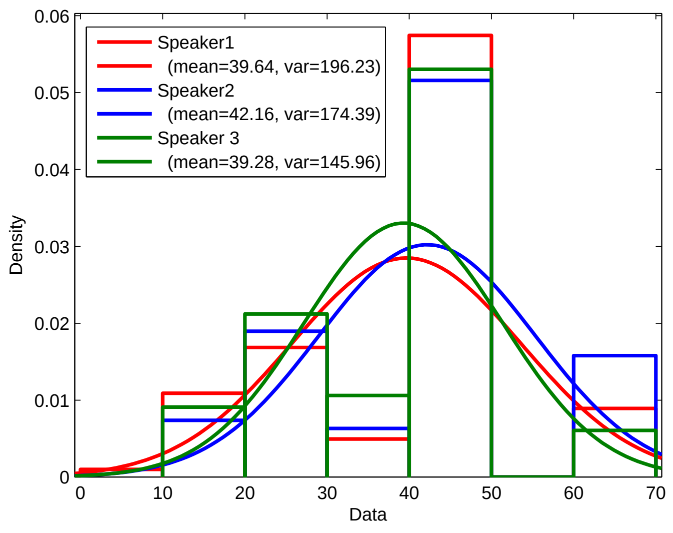

Figure 7.1: Histogram of the packet-lengths of letters ”P I T T S B U R G H” spoken by different speakers

replaced by the Secure Real Time Transport Protocol (SRTP) [117]. SRTP standardizes only a single encryption algorithm namely the Advanced Encryption Standard (AES) which can be used in two cipher modes: segmented integer counter mode and f8 mode. In the former mode, which is the default encryption mode, an integer incremental counter is used with the default encryption key length of 128 bits and a default session salt key of length 112 bits. The f8 mode is a variation of the output feedback mode with a more elaborate initialization and feedback function. It uses the same default sizes for session key and salt as the AES segmented integer counter mode. As clearly mentioned in [117] “none of the pre-defined encryption transforms uses any padding; the RTP and SRTP payload sizes match exactly.” Hence the packet length information remains unchanged after encryption and all exploitation techniques based on this information remain as valid after encryption as they were before encryption.

7.4 Problem Statement

The break up of the speaker recognition as a forensic investigation problem is given by the following two subproblems.

7.4.1 Speaker Identification

Given a set of n suspected speakers $\{S_1, \cdots, S_n\}$ and a disputed anonymous encrypted speech segment O, the investigator is asked to identify the speaker $S \in \{S_1, \cdots, S_n\}$ of the anonymous speech segment. The identified speaker is one which gives the maximum value for $p(S|O)$. It is assumed that the investigator has access to m speech segments $V = \{v_1, \cdots, v_m\}$ of each suspect and the true speaker is assumed to be one of the suspected speakers.

7.4.2 Speaker Verification

Given an observation sequence in the form of a speech segment O, and a hypothesized speaker S, speaker verification is a basic hypothesis test between H_0 (O is from the hypothesized speaker S) and H_1 (O is not from the hypothesized speaker S.) If $\frac{p(O|H_0)}{p(O|H_1)} \geq \theta$, we accept H_0, otherwise reject H_0 (accept H_1) where $p(O|H_i), i = 0, 1$ is the probability density function for the hypothesis H_i evaluated for the observed speech segment O and θ is the decision threshold for accepting or rejecting H_0.

7.5 Proposed Approach

Fig. 7.2 shows an overview of our proposed approach. First, the packet length information from the encrypted VoIP conversation are extracted and used to create suitable models for the different speakers. The unknown communication can then be classified, using some classification/regression techniques on the basis of the pre-trained models for each speaker. These steps are further explained throughout the rest of this section.

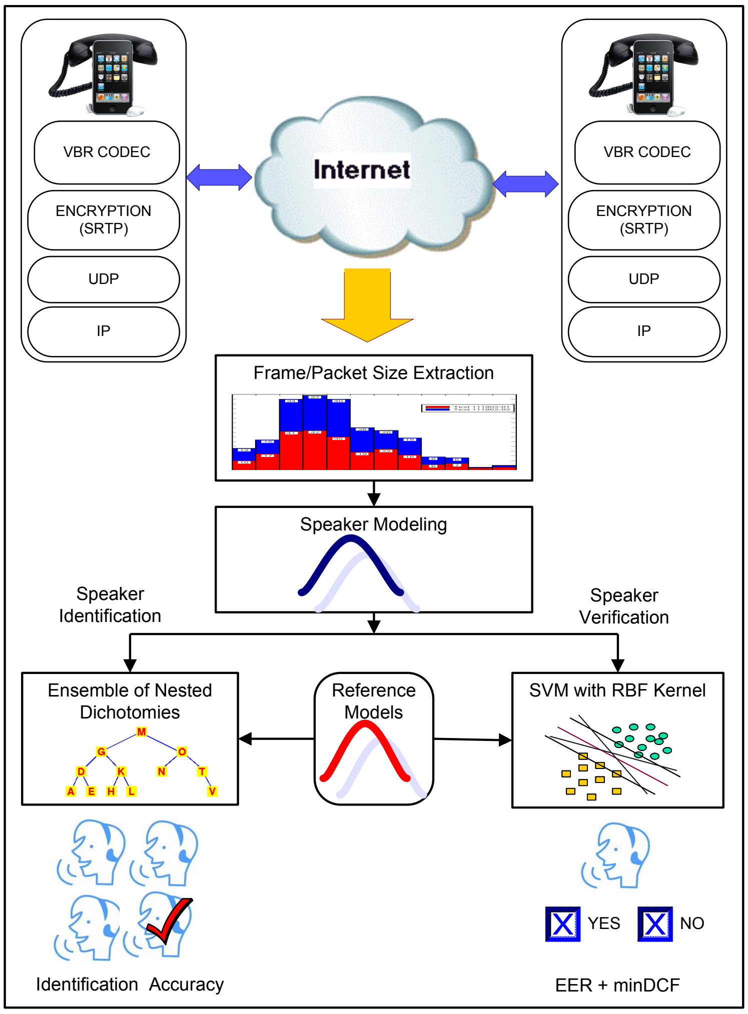

Figure 7.2: Overview of the proposed approach for speaker identification from encrypted VoIP communications

7.5.1 Packet Length Extraction

The packet length information can be extracted from the encrypted VoIP conversation using one of the numerous open source packet sniffing tools or the techniques mentioned in [119, 95]. In addition, multimedia container formats like the Ogg [124] and MPEG-4 [125] are also used for carrying multimedia information in packets over the Internet. For the voice transmission over packet switched networks, Ogg uses the Speex encoder which handles voice data at low bit rates

(8-32 kbit/s/channel) and the Vorbis encoder which handles general audio data at mid to high-level variable bit rates (16-500 kbit/s/channel) [122]. To cater for the VBR encoding mechanism, the Ogg page headers reserve a specific field in the page header as segment table. On one hand, it provides for efficient multiplexing and seamless integration of multimedia data independent of the underlying compression and encoding mechanisms. On the other hand, the segment length information can be retrieved from the page header without even inspecting the internal packet contents.

7.5.2 Modelling and Classification

For modelling the speaker's identity with respect to the variable packet-length information, various modelling approaches are explored. In case of conventional speaker recognition approaches, the observations on which the models are based are multidimensional vectors generally in the form of Cepstral features extracted per frame [102]. In the encrypted VoIP scenario, we only know the length of the frame, or for that matter the packet, which is a scalar value per frame. We used discrete hidden Markov models (HMMs) to create a model for each speaker based on the sequence of packet-length information. An HMM is characterized by two model parameters, M and N, and three probability measures A, B and π, where N denotes the number of states $\{S_1, \cdots, S_N\}$ in the model, M denotes the number of distinct observation symbols per state, A is the state transition probability distribution, B is the observation symbol probability distribution and π is the initial distribution. HMMs use the Baum Welch expectation maximization algorithm [126] for calculating the model parameters and the Viterbi search [127] for finding the most likely sequences of hidden states.

The other approach which is also used in [119] for language identification is to create tri-gram models out of the frame lengths for each speaker. Using trigram probabilities, speakers were modelled as GMMs with different model orders. Mathematically, a GMM $\lambda(y)$ is given by [118]:

$$\lambda(y) = \sum_{i=1}^{M} \alpha_i G(y; m_i; \Sigma_i) \tag{7.1}$$

where $G(y; m_i; \Sigma_i)$ is a Gaussian model with mean m and covariance Σ. The weight of each mixture is represented by α_i and M denotes the total number of Gaussians. Different Expectation Maximization algorithms are used to find the optimum values of mixture weights [128]. Each Gaussian in the mixture has its own mean and covariance matrix that has to be estimated separately [129].

In addition to the HMMs and GMMs described above, we have also tested several other classifiers. The classifier which obtained the best accuracies for speaker identification was based on the Ensemble of Nested Dichotomies (END) [130] which is a recently introduced statistical technique for tackling multi-class problems by decomposing it into multiple two-class problems. Using C4.5 decision tree [131] and logistic regression [132] as base learners, the ensemble of nested dichotomies shows better classification accuracies compared to the case where we apply these learners directly to multi-class problems. The probability estimates produced by binary classifiers are multiplied together, considering these to be independent, in order to obtain multi-class probability estimates. Nested dichotomies can be represented as binary trees. At each node, we divide the set of classes A associated with the node into two subset B and C that are mutually exclusive such that B and C together contain all the classes in A. The nested dichotomies root node contains all the classes of the corresponding multi class classification problem. Let L_{i1} and L_{i2} be the two subsets of class labels produced by a decomposition of the set of classes L_i at internal node i of the tree and let $p(l \in L_{i1} | y\ ,\ l \in L_i)$ and $p(l \in L_{i2} | y\ ,\ l \in L_i)$ be the conditional probability distribution estimated by the two class model at node i for a given instance y. Then the estimated class probability distribution for the original multi-class problem is given by:

$$\begin{aligned} p(l = L|y) = \prod_{i=1}^{n-1} (I(l \in L_{i1})p(l \in L_{i1}|y, l \in L_i) \\ + I(l \in L_{i2})p(l \in L_{i2}|y, l \in L_i)) \end{aligned} \tag{7.2}$$

where $I(\cdot)$ is the indicator function and the product is over all the internal nodes of the tree.

In the speaker verification domain, various classification and modelling approaches can be applied in order to determine whether a particular speech utterance belongs to a particular speaker or not. We used several classification techniques to verify the speaker identity using the tri-grams of the packet lengths of the speech encoded by VBR encoding mechanism. One way to deal with the speaker verification problem is to consider it as a two-class classification problem and assign the probability of the likelihood of the model as the true and imposters scores in order to calculate the equal error rates and minimum detection cost function. Another technique which we also used is to tackle the speaker verification problem as a regression problem [133] by assigning different numerical values to true speakers and imposters during the training process. Then we set a threshold for the binary decision. For any unknown utterance, we calculate the values via the same regression modelling technique and attribute the utterance to the target speaker depending upon the threshold and the calculated value. The classifier we used for this approach is the SVM with RBF kernel [134] [135].

7.6 Experimental Evaluation

Our experiments were conducted on the CSLU speaker recognition database [136] which consists of speech files from 91 speakers recorded in twelve sessions over a period of two years. Each session comprises of 96 files each of 2 to 20 seconds duration. We first encoded the wave files of the database with Speex using variable bit rate encoding in narrow band mode. This encodes the files with eight different bit rates depending upon the speech contents in each 20 msec frame. The VBR encoder encoded the speech files with eight distinct bit rates resulting into different frame sizes each of 6, 10, 15, 20, 28, 38, 46 and 62 bytes. The sequence of these frame sizes depends on the content as well as acoustics of the underlying speech. A Matlab application was developed for extracting the packet-length information from the encrypted files. Various techniques were then employed to correlate the frames sizes with the speaker identity.

7.6.1 Speaker Identification

In case of speaker identification, the job of the forensic investigator is to identify the potential perpetrator in a group of potential suspects. The number of potential suspects may vary from one case to another. We conducted experiments on groups of 5, 10, 15 and 20 suspects. Various modelling and classification approaches were used for the experimental evaluation of identifying the potential perpetrator from the packet length information extracted from encrypted VoIP conversations. The following are the worth mentioning experiments.

HMM Tests: The HTK toolkit [**?**] was used to develop HMM models, for different speaker based on their corresponding sequence of frame sizes. 300 speech files per speaker were used for training and 100 files were used for testing. The HMM modelling approach achieves an identification accuracy of 54% for a group of ten speakers.

GMM Tests: For the GMM tests, we calculated the tri-gram probability of the eight symbols (frame lengths) thereby creating a 512 component vector for each speech file. For example, the first component of the the 512 component vector corresponds to the probability of the sequence (6,6,6) in the sequence of symbols, the second component to (6,6,10) and so on. These probability distribution of the tri-grams of each speaker were modelled as a GMM. Again we used 300 files per speaker for training and 100 files for testing. Various model orders, from 8 to 64 were used, but no significant improvement could be observed with increase in models orders. The speaker identification accuracies obtained through the GMM approach showed slightly better results as compared to the HMMs. For example the identification accuracy in case of 10 speakers increased form 54.2% to 58.9% in case of GMMs with model order 16.

Bayes Net and ENDs Tests: The WEKA tool kit [137] was used to investigate the performance of different types of classifiers and regression techniques in our speaker identification problem. Among various classification techniques available, the Bayesian Network classification technique [138] showed results better than both HMMs and GMMs. For the training data and test data we used the tri-gram probability as features and the speakers identity as classes. We conducted experiments using 10 fold cross validation evaluation to avoid any biases in the data. Furthermore, the use of meta classification has showed significant improvement over simple classification techniques. In this case, the discriminative power of various base classifier is combined to achieve better classification accuracies. We conducted some tests using various meta classifiers with different combinations of base classifiers. The Ensemble of Nested Dichotomies (ENDs) showed significant improvement over all other classification methods. For the training and testing data we again used the tri-gram probability distributions of the frame length sequencing to compare the accuracies with the same data set and features.

7.6.1.1 Results

Table 7.1 shows a summary of the identification accuracies of the above mentioned four significant classification techniques on a group of 5, 10 , 15 and 20 speakers.

Table 7.1: Speaker Identification Accuracies of Various Classification Methods

	Identification Accuracies (%)			
No. of Speakers	**HMM**	**GMM**	**BayesNet**	**ENDs**
5	60.5	63.7	68.4	74.9
10	54.2	58.9	59.2	72.4
15	43.7	41.7	51.7	59.8
20	38.4	39.6	43.2	51.2

We also measured the *precision* and *recall* of all the above proposed techniques. The average values of the precision, recall for 10 speakers obtained throughout the above mentioned four classification techniques are presented in Table 7.2. The variation of F-measure, which is defined as harmonic mean of precision and recall, with number of speakers with respect to the classification techniques is depicted in Fig. 7.3.

Table 7.2: Average Precision, Recall and F-measure values of speaker identification on a group of 10 speakers

Classification Technique	Precision	Recall	F-Measure
HMM	0.557	0.542	0.553
GMM	0.599	0.589	0.591
Bayes Net	0.601	0.592	0.598
ENDs	0.734	0.724	0.723

7.6.2 Speaker Verification

Speaker verification can be thought of as a two class classification problem. We conducted experiments on the same data set of speakers as the speaker identification but in this case we first used

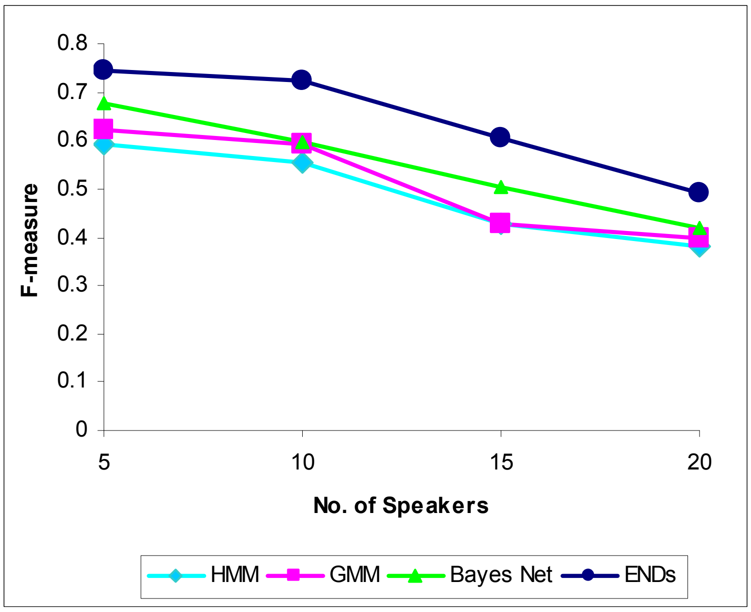

Figure 7.3: F Measure Variation with Number of Speakers and Classification Techniques

classification techniques to model two classes, one for the target speaker and one for the cohort or background model. Two types of cohort models were created, one for the male group and another one for the female group. The two cohort models were trained using the complete data sets of male and female speakers respectively. For the target speaker, we used four hundred files per speaker. For evaluation of our classification methods, we used the 10 fold cross validation approach to avoid any biases in the evaluation experiments and to judge the classification method over the entire database reserving 90% for training and 10% for testing. The NIST toolkit was used to calculate the equal error rates. For the true speakers and imposters' scores, we used the probability scores of our classifier as input.

7.6.2.1 Speaker verification via classification

For classification, we used three different classification techniques, namely Adaboost.M1 [139], Discriminative Multinomial Naive Bayes (DMNB) [140] and Bayesian Network [141] classifiers. Fig. 7.4 shows a typical DET plot of one speaker from our database using the three classification techniques. The point on the DET plot which gives the minimum cost detection function is marked on each curve.

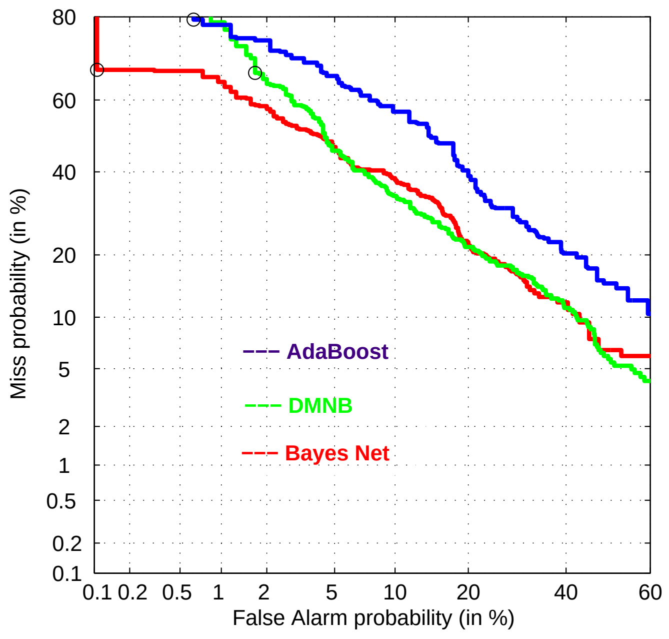

Figure 7.4: A typical DET plot of speaker verification with different classification and regression techniques

7.6.2.2 Speaker Verification via Regression

In this case, we again used three different regression techniques. These are linear regression [137], SVM with Sequential Minimum Optimization (SMO) [142] and SVM with RBF kernel [134]. We used regression scores of the true speakers and imposters for calculating the equal error rates and minimum detection cost function. The regression approach via SVM with RBF kernel produced lower EER as compared to the Bayesian Network classifier. Fig. 7.5 shows the typical DET plots of one randomly selected speaker from our database when using the three regression techniques described above.

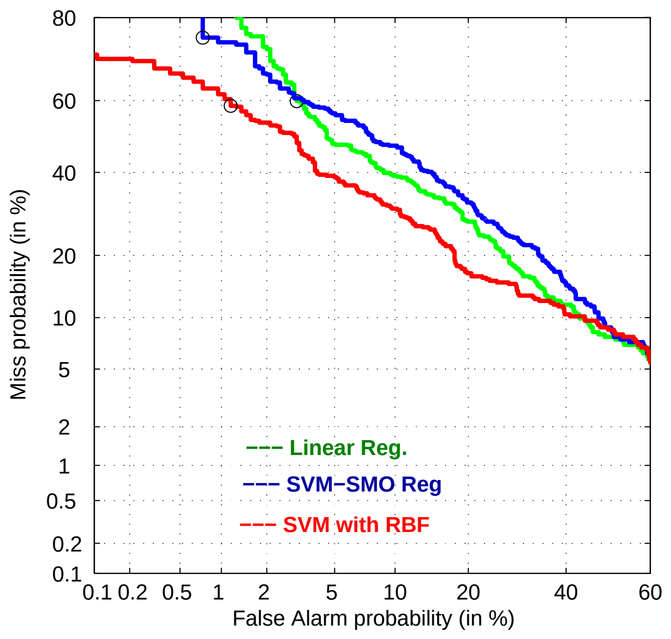

Figure 7.5: A typical DET plot of speaker verification with different classification and regression techniques

7.6.3 Discussion

Although the identification and verification accuracies achieved in our experiments are not comparable to the ones achieved in the un-encrypted speech domain, yet these are by far superior to random guessing as one would expect in case of encrypted communication. In the speaker identification domain, the results obtained on modeling the packet lengths as discrete HMMs could not achieve very encouraging results. The reason for this could be that since HMMs are based mainly on the temporal sequencing of the observations. They do not depend much on the overall average statistics of the modeled signal. In the frame length sequencing information since the total number of discrete symbols is very small (just eight) hence large similarities in the temporal sequencing of these symbols between different speakers could be observed.

In order to cater for the average statistics of the signal to be modeled, we switched to Gaussian mixture modeling approach. This time we did not use the temporal sequencing of the underlying discrete symbols as input features to our models. Instead, we created tri-gram probability distributions to incorporate the the overall signal statistics along with temporal sequencing. This time we could achieve comparatively better identification accuracies validating our premise. We observed that the probability distributions are very sparse and for most of the tri-gram sequences like 6,6,62 and 10,10,6 etc we could not observe even a single occurrence in the complete data set. One way to address this issue is to use smoothing techniques used by the natural language processing community. We reserve this task for future work. For the time being we concentrated on other modeling and classification techniques. Moreover, from the insignificant improvement in results with increase in model orders in case of GMMs, we derived that the distribution of the packet lengths cannot be distinctively attributed to different speakers using Gaussian Mixtures with distinct Gaussian weights, means or covariance matrices for each speaker. The Bayesian Network which is basically probabilistic graphical model that represents a set of random variables and their conditional independencies via a directed acyclic graph. The conditional probabilities of the packet length sequences mapped better via the Bayesian Network classifier to the speakers'identity as compared to HMMs and GMMs. The result shown by the ensemble of nested dichotomies suggest that this problem can be best solved by the combination of different classification techniques. Using the C4.5 decision tree algorithm with logistic regression as base learners, the meta classifier could achieve comparatively better recognition accuracies suggesting that this problem can be better solved through binary classification methods applied over multi-class problems. As far as the variation of recognition accuracies with increasing number of speakers is concerned, this is in perfect synchronism to any conventional multi-class classification problem where the recognition accuracies decline with increasing number of classes.

In the speaker verification domain, which is a binary classification problem, the regression techniques have shown better EER and minDCF as compared to the classification techniques. The accuracy in case of speaker verification depends on how the two predicted scores of the two classes are distinctly apart. This helps in choosing a suitable threshold for the binary decision. The output of a classifier is the predicted class label with the probability score. In case of regression, the classifier predicts the score directly. This gives us more freedom to choose the scores for the true speaker and the background model. Moreover, since SVMs generally perform better in two class classification problem, the general observation remained valid in our case also.

Table 7.3 shows the mean EER and minimum CDF obtained when using the above discussed classification and regression methods.

Table 7.3: Speaker Verification Accuracies of Various Classification Methods

	Classification			***Regression***		
Verification	A.Boost	DMNB	Bayes	SVM-SMO	Lin. Reg	SVM-RBF
EER(%)	23.4	20.1	19.5	22.3	21.3	17.1
minDCF	0.0856	0.0838	0.0690	0.0901	0.0830	0.0681

7.7 Summary

With the advancement in voice over IP applications in which the un-encrypted speech communication is diminishing, access to un-encrypted speech can prove to be a very difficult task for investigators. Therefore, future forensic applications need to look into the possibility of identifying perpetrators from encrypted speech segments. This paper is an endeavor in this direction. Several techniques for forensic speaker recognition from encrypted VoIP conversations were presented. It has been shown that the variable packet-length information in case of variable bit speech encoding mechanism can be exploited to extract speaker dependent information from encrypted

VoIP conversations. Although the identification and verification accuracies achieved in our experiments are not comparable to the ones achieved in the un-encrypted speech domain, these are by far superior to random guessing as one would expect in case of encrypted communication. It should also be noted that while the current state of the art speaker recognition techniques have not matured enough to be produced in a court as the sole source of evidence against a suspect, these techniques are nonetheless valuable tools that can facilitate forensic investigations. In the same context, the computational data obtained by our experiments, obviously, cannot be used as a complete forensic evidence. However, together with other sources of evidence they can provide some clues for further directions to the forensic investigators.

Chapter 8

CONCLUSION

8.1 Introduction

In this chapter we summarize the contributions and results presented in this thesis and deliberate on some of the issues which may be taken up as a future work. The concept of using linear complexity for distinguishing between the speech and silence zones in an encrypted conversation has been presented. The two time pad situation has been discussed in the literature and all the exploitation techniques presented so far concern the text based data. In this thesis the cryptanalysis of two time pads of encrypted speech signals is also presented for the first time. The idea has been discussed keeping in view almost all the modern speech coding and compression techniques. The idea of speaker recognition from partially encrypted speech has been introduced for the recently introduced selective encryption techniques for mobile multimedia applications. Since all modern speech communication are switching to voice over IP, thereby discussing speech without referring to voice over IP would be an incomplete correspondence. Most of the simulations presented in the thesis are coded in MATLAB , C-language and Perl Scripting Language. Since some of the ideas are presented for the first time, therefore room for detailed simulations in different experiments in varied situations still exists.

8.2 Contributions of this Thesis

The contributions presented in this thesis can be summarized as under:

- The statistical properties of digitized speech signals are different from text and other forms of data. We also present techniques in this thesis to exploit these statistical properties of speech signal for ciphertext only cryptanalysis of stream ciphered digitized speech both in the single and multiplexed communication environment.

- A specific situation of stream ciphered transmission of data known as the two time pad situation has been discussed at length with respect to text based data encoded through conventional encoding mechanism. The exploitation techniques in this situation for speech data has been introduced for the first time in this thesis. Both the parameter and waveform encoded speech signals have been discussed at length.

- The concept of language modeling has been extended to inter frame speech parameters modelling the idea presented for the first time by us. All the parameters present in the speech frames have been found to be non uniformly distributed except for the fixed code book pulse signs which are uniformly distributed between positive and negative signs.

- The recently introduced partial encryption techniques for mobile multimedia applications have been analyzed from the perspective of speaker's identity protection.

- We are the first, to the best of our knowledge, to apply speaker identification and verification to *encrypted* VoIP conversations.

- The recently developed container formats which are used to store and carry multimedia information over the Internet are explored from the perspective of speaker recognition in case of encrypted communications.

- Our experimental results on speaker recognition from encrypted VoIP conversations indicate that different types of classification and regression techniques, that are usually used in data mining and machine learning applications, outperform Gaussian mixture models and hidden Markov models-the classifiers which perform very accurately in the conventional speaker recognition studies.

8.3 Future Work

Following are some of the open problems for research in the research work presented in this thesis:

- Techniques other than linear complexity for the purpose of classifying the silent and speaking zones in the cipher text presented in Chapter 2 also needs to be looked into. In case of very large linear complexities the, i.e. if the linear complexity is larger than half the length of the silent zone then the linear complexity analysis test will fail. The asymptotic analysis of the classification of silence zones and noise zones as well as that of the cryptanalysis on the basis of this technique also needs to be investigated.

- The concept of context-dependent tied-state multi-mixture tri-phones has shown significantly better performance from speech recognition point of view. The same concept when applied to cryptanalysis of two time pads in case of stream ciphered digitized speech using ASR techniques is expected to show better accuracy of decoded phoneme pairs.

- The inter frame speech parameters modeling idea presented in Chapter 6 can be extended to lossless compression using an adaptive lossless compression techniques e.g. Adaptive Hufman coding since the all the parameters are non uniformly distributed from inter frame point of view except for Fixed Code Book Pulse Signs which are uniformly distributed between positive and negative pulses.

- The selective encryption technique discussed in Chapter 6 is shown to reveal the speakers identity, modification of the technique can be looked into to avoid this disclosure. Language identification from the partially encrypted speech can also be taken up as a future work.
- Modifications in the SRTP protocols can be suggested to avoid exploitation of packet length information for revelation of speaker's identity.

Appendices

Appendix A

SOURCE CODE FOR PARAMETER ENCODED SPEECH DECODING

A.1 Matlab Code for XOR of two Wave Files

```
clear all

[x_filename,x_filepath] = uigetfile('*.wav','Open File X');

fid1=fopen(strcat(x_filepath,x_filename),'r');

a=fread(fid1, 'uint16');

[y_filename,y_filepath] = uigetfile('*.wav','Open File Y');

fid2=fopen(strcat(y_filepath,y_filename),'r');

b=fread(fid2, 'uint16');

a_s = size(a);

b_s = size(b);

if (a_s(1) > b_s(1))    % b is smaller in size

    c = bitxor(b,a(1:b_s));

    c(1:512)=b(1:512);

else                    % a is smaller in size

    c = bitxor(b(1:a_s),a);

    c(1:512)=a(1:512);

end

c=bitxor(b(1:11812),a);
```

```
c(1:512)=a(1:512);

x_s = size(x_filename);

y_s = size(y_filename);

c_filename = strcat(x_filename(x_s(1,1):x_s(1,2)-4),'_',

y_filename(y_s(1,1):y_s(1,2)-4),'.wav');

fid3=fopen(strcat('C:\simulations\wavxor\',c_filename),'w');

fwrite(fid3,c,'int16');

msgbox('XOR of file X and Y has been saved.','warn')

fclose all;

clear all;
```

A.2 MATLAB Code for Transitional Probability Calculations with Simple Back Off Smoothing

```
clear all

Lamda = 0.5;

% Prob(1-->2)= lamda * P(1-->2) + (1-lamda)* P(2)

P = LSFIndex1;

% the data on which the bigram prob is being calculated

l_P = length(P);

l_T = 1024;

% No of alphabets in the data

T_count = zeros(l_T,l_T);

Trans_prob = zeros(l_T,l_T);

for (m=1:l_T)
```

```
    Index = find (P==m-1);
    % find the locations of m in P
    l_Index = length(Index);
% gets the total number of instances where
% m is occuring in P
    if (l_Index > 0)
    % check if no of instances is greater than 0
        for (n=1:l_Index)
% iterate for the total number of items in index
            if ((Index(n)+1)<=l_P)
            % check if index for P,
%(Index(n)+1)we are not going beyond the lenght of P
                i = P(Index(n))+1;
                j = P(Index(n)+1)+1;
                T_count(i,j) = T_count(i,j)+1;
                % increment that i->j has occured
            end
        end
    end
% calculate the probabilities of the row number m
    Trans_prob(m,:) = (1+T_count(m,:))/(l_T + l_Index);
    %length of T and 1 added to cater for
    %smoothing in case of zero entries
    for(k=1:l_T)
```

```
        Trans_prob(m,k) = (Lamda*Trans_prob(m,k)) +
        ((1-Lamda)*UniProb(k));
        % To cater for back off smoothing
    end
    % T_count of row m is complete
end
clear Index T_count P i j l_Index m n Index
l_P l_T Lamda k;
```

A.3 MATLAB Code for Transitional Probability Calculations with Witten Bell Smoothing

```
N=1;
% here N is the number of unique entries that follow
% the history i.e. P(1-->2)  = {Count(1-->2) +
% N1 * UniProb(2)}/{N1 + Count(1-->?)} where N1
% is the number of unique entries that follow 1.
P = LSFIndex1;
% The data on which transitional prob is calculated
l_P = length(P);
l_T = 1024; % No of alphabets in the data.
T_count = zeros(l_T,l_T);
Trans_prob = zeros(l_T,l_T);
for (m=1:l_T)
    Index = find (P==m);
    % find the locations of m in P
    l_Index = length(Index);
% gets the total number of instances where
% m is occuring in P
    if (l_Index > 0)
    % check if no. of instances is greater than 0
        for (n=1:l_Index)
    % iterate for the total number of items in index
            if ((Index(n)+1)<=l_P)
% check if index for P,
%(Index(n)+1)we are not going beyond the length of P
                i = P(Index(n));
                j = P(Index(n)+1);
                T_count(i,j) = T_count(i,j)+1;
                % increment that i->j has occurred
            end
        end
        N=length(find(T_count(m,:))) ;
        %find the number of non zero entries in row m
        %which is the number of unique entries
        %that follow m
    else
        N=1;
    end
    % calculate the probabilities of the row number m
```

```
      %Trans_prob(m,:) = (1+T_count(m,:))/(l_T + l_Index);
      %length of T and 1 added to cater for smoothing
      %in case of zero entries
    for(k=1:l_T)
        Trans_prob(m,k) = (T_count(m,k)
        + N * UniProb(k)) / (N + l_Index); % To cater
        %for back off smoothing
    end
    % T_count of row m is complete
end
clear Index T_count P i j l_Index m n Index l_P l_T k N
```

A.4 MATLAB Code for Customized Viterbi Decoding

```
% complete implementation of Decoding
% for values ranging between 1 and 2^n
% where n is the number of bits representing the value
% Seq = input sequence
% T = Transition Matrix
% L = log probability transition matrix
% S1,S2..  states
% S1S2, S2S3,... weights of edges from
% the corresponding stages
% R1 = L + S1S2 Resultant wight matrix of Stage 1 to Stage 2
% W1_2 Min log prob paths from stage 1 to 2
% Short_Node1_2 index of node at stage 1 with the
% shortest path to stage 2
clear all
% initialization
%load('C:\Documents and Settings\Liaqat
%\Desktop\sim\Sequence.mat');
Seq = data;
l_Seq = length (Seq);
%T=[0.364 0.273 0.182 0.09; 0.103 0.241 0.413 0.241;
0.115 0.385 0.269 0.231; 0.05 0.3 0.25 .3];
T=Trans_prob;
size_T = 1024; %nter the number of alphabets
%to cater for the zeros in transition
%probability matrix since log of zero
%is not defined
% for i=1:4
%     for j=1:4
%         if (T(i,j)==0)
%             T(i,j)=0.00001;
%         end
%     end
% end
L = -log(T);
%if using log prob
```

```
% L=T;

%If not using log prob

%Initializing Variables

S1 = zeros(1,size_T);

S2 = zeros(1,size_T);

S1S2 = zeros(size_T);

R = zeros(size_T);

R_t = zeros(size_T);

W = zeros(1,size_T);

W_Prev = zeros(1,size_T);

% if using log probabilities

% W_Prev = ones(1,size_T);

% if not using log probabilities

Short_Node = zeros(1,size_T);

% Outer loop depending upon the sequence length

for (loop = 1:l_Seq-1)

    % for calculating S1

    for i=1:size_T

        S1(i) = bitxor(i-1,Seq(loop));

        S2(i) = bitxor(i-1,Seq(loop+1));

    end

% for calculating weights of edges

    for (i=1:size_T)

        for (j=1:size_T)
```

```
            S1S2(i,j) = L (S1(i)+1,S2(j)+1);
        end
    end
% Resultant matrix
    R_t = L + S1S2;
    % if using log probabilities
%     R_t = L .* S1S2;
% if not using log of probabilities
    for (i=1:size_T)
        R(i,:) = R_t(i,:) + W_Prev(i);
        %if using log probabilities
%         R(i,:) = R_t(i,:) * W_Prev(i);
% if not using log probabilities
    end
    % min/max weight edges
    %(min for log prob and max for prob without log)
    W = min(R);
    % If using log probabilities
%     W = max(R);
%if not using log probabilities
    % to find index of the shortest node to S2
    for (i=1:size_T)
        [tt,j] = find (R(:,i)==W(i));
        Short_Node(i) = tt(1);
    end
    % to update the edges weight of the previous
    % stage to be added to the new minimum weights
    W_Prev = W;
    %writing shortest edges from different
    % nodes at each stage to file
    name = strcat('C:\Documents and Settings\Liaqat
    \Desktop\sim\ShortNode\SN',num2str(loop),'.txt');
    fid = fopen (name,'w');
    fwrite(fid,num2str(Short_Node));
%     if (loop == l_Seq-1)
%         fwrite (fid,'  ');
%         fwrite (fid,num2str(W));
%     end
    fclose(fid);
end
% end of loop
% Back Tracking the shortest path
% and reading off the labels
% P and Q are the arrays such that P xor Q = Seq
P = zeros(1, l_Seq);
Q = zeros(1, l_Seq);
% back tracking the last node. We already have the W
% and Short_Node of the last stage
```

```
P_temp = find(W == min(W)) - 1;
% if using log probabilities
% P_temp = find(W == max(W)) - 1;
% If not using log probabilities
P(l_Seq) = P_temp(1);
Q(l_Seq) = bitxor(P(l_Seq),Seq(l_Seq));
for (i = l_Seq-1:-1:1)
    name = strcat('C:\Documents and Settings\Liaqat
    \Desktop\sim\ShortNode\SN',num2str(i),'.txt');
    fid2 = fopen(name,'r');
    Short_Node = fread(fid2,'uint8=>char');
    %read Short_Node as character string
    fclose(fid2);
    % convert to double
    Short_Node = str2num(Short_Node');
    % Calculate P & Q
    P(i) = Short_Node(P(i+1)+1)-1;
    Q(i) = bitxor(P(i),Seq(i));
end
fclose all;
clear fid fid2 i j l_Seq loop name size_T tt
W W_Prev Short_Node Seq S1 S2 S1S2 R_t R P_temp L T ans;
```

Appendix B

SOURCE CODE FOR PARTIALLY ENCRYPTED SPEECH CRYPTANALYSIS

B.1 Matlab Code for Extracting the Non Encrypted Portion from the CELP Frame

```
%This file extracts the non encrypted portion
%from the 80 bit CELP frame
clear all;
fid=fopen('D:\CMUexp\encoded\train\mcen\an129-mcen-b','rb');
% to open the file that contains the CELP frames
A=fread(fid);
A=uint8(A);
L=length(A);
for (i=0:L/10-1)
    L0(i+1)=bitget(A(i*10 +1),8);  j=i*10+2;
% to get the 2nd byte of every frame
%(each frame is 10bytes(80 bits long))
    x=bitand(A(j),7);
% to get the last three bits
    x1=bitshift(x,2);
% to append two zeros at the right to make it 5 bits
```

```
    y=bitshift(A(j+1),-6);
% to get the two left most bits of the 3rd byte
    L3(i+1)=bitor(x1,y); k=i*10+4;
% to get the 4th byte of every frame
    xx=bitand(A(k),31);
% to get the last five bits
    xx1=uint16(xx);
%to make it 16 bit integer
    xx2=bitshift(xx1,8);
    C1(i+1)=bitor(xx2,uint16(A(k+1)));
    l=i*10+6;
% to get the 4th byte of every frame
%(each frame is 10bytes(80 bits long))
    S1(i+1)=bitshift(A(l),-4);
% to get the ist four bits
    m=i*10+8;
% to get the 8th byte of every frame
    xxx=uint16(A(m));
% to get the 8th byte in 16 bit format
    xxx1=bitshift(xxx,5);
%to make it the most significant bits in 13 bit integers
    yyy=bitshift(A(m+1),-3);
%to get the next five bits of the 13 bit integer
    C2(i+1)=bitor(xxx1,uint16(yyy));
    n=i*10+9;
% to get the 8th byte of every frame
    xxxx=bitand(A(n), 7);
% to get the last three bits
    xxxx1=bitshift(xxxx,1);
%to make room for the fourth bit
    yyyy=bitshift(A(n+1),-7);
%to get the most significant bit
    S2(i+1)=bitor(xxxx1,yyyy);
end
fclose(fid);
numsample= L/10;
num1=bitshift(numsample, -8);
% to get the eight most significant bits
num2=bitand(numsample,255);
% to get the eight least significant bits
%NOTE: The numsamples should be less than 2^16 (65536).
%For higher numbers change the leading two zeros accordingly.
header= [0 0 num1 num2 0 1 134 160 0 24 0 9];
% for the header, the first four bytes correspond to the
%binary representation of the number of vectors in the
%file eg [0 0 17 47] corresponds to 4399. keep the other
%components as it is.
for p=0:L/10-1
```

```
    q=6*p+1;
    V(q)=single(L0(p+1));
    V(q+1)=single(L3(p+1));
    V(q+2)=single(C1(p+1));
    V(q+3)=single(S1(p+1));
    V(q+4)=single(C2(p+1));
    V(q+5)=single(S2(p+1));
end
fid=fopen('D:\CMUexp\reduced\train\mcen\an129-mcen-b','a');
%To open the file to write the reduced CELP vector into.
fwrite(fid,header);
fwrite(fid,V,'single', 'b');
% single means floating point with 32 bits and
%'b' represents the IEEE 754 floating point format
fclose(fid);
clear A i j k l m n x xx1 xx2 xxx xxxx xxxx1 x1 y xxx1
yyy yyyy fid xx L p q header num1 num2 numsample;
```

B.2 Matlab Code for Converting CELP files to HTK Recognizable Format

```
%This file converts the CELP encoding to
%HTK Recognizable Format
clear all;
fid=fopen('D:/OGIPartial/encoded/1022/10221aa1.raw.celp','rb');
A=fread(fid);
```

```
A=uint8(A);

L=length(A);

for (i=0:L/10-1)

L0(i+1)=bitget(A(i*10 +1),8);

    j=i*10+2;

    x=bitand(A(j),7);

    x1=bitshift(x,2);

    y=bitshift(A(j+1),-6);

    L3(i+1)=bitor(x1,y);

    k=i*10+4;

    xx=bitand(A(k),31);

    xx1=uint16(xx);

xx2=bitshift(xx1,8);

    C1(i+1)=bitor(xx2,uint16(A(k+1)));

    l=i*10+6;

    S1(i+1)=bitshift(A(l),-4);

m=i*10+8;

    xxx=uint16(A(m));

    xxx1=bitshift(xxx,5);

    yyy=bitshift(A(m+1),-3);

    C2(i+1)=bitor(xxx1,uint16(yyy));

    n=i*10+9;

    xxxx=bitand(A(n), 7);

    xxxx1=bitshift(xxxx,1);
```

```
yyyy=bitshift(A(n+1),-7);
    S2(i+1)=bitor(xxxx1,yyyy);
end
L0=double(L0);
L3=double(L3);
C1=double(C1);
S1=double(S1);
C2=double(C2);
S2=double(S2);
sumL0=0;
sumL3=0;
sumC1=0;
sumS1=0;
sumC2=0;
sumS2=0;
for j=1:length(L0)
    sumL0=sumL0+L0(i);
    sumL3=sumL3+L3(i);
    sumC1=sumC1+C1(i);
    sumS1=sumS1+S1(i);
    sumC2=sumC2+C2(i);
    sumS2=sumS2+S2(i);
end
avgL0=sumL0/length(L0);
```

```
avgL3=sumL3/length(L3);

avgC1=sumC1/length(C1);

avgS1=sumS1/length(S1);

avgC2=sumC2/length(C2);

avgS2=sumS2/length(S2);

X=[avgL0; avgL3; avgC1; avgS1; avgC2; avgS2];

fid1=fopen('D:/OGIPartial/arff/test.arff','a')

fprintf(fid1,'%d, %d, %d, %d, 1022\n\n',X);

fclose all;
```

B.3 MATALB Code for Speaker Verification Decision

```
%This file extracts the log probabilities from the

%rec.mlf file and generates data for speaker

%verification decision

clear all;

close all;

fid=fopen('D:\SV2\recout\trainingdata\mdcs2.mlf', 'rt');

% Open the file containing the results from HTK.

C = textscan(fid,'%s %s %s %f','headerlines',2,...

'commentStyle',{'"','"'});

% Read the results into a four column cell array C.

A=C{3};

% Take the third cell which is either the speaker

% or reference.
```

```
A=A';

X=cell2mat(A);

% Matlab cannot manipulate the data in cell format,

% hence it has to be changed to matrix fmt.

B=C{4};

% Take the log prob values into B.

j=1;

m=1;

k=0;

sum=0;

var=0;

for i=1:+2:length(A)-1

    Y=X(4*k+1:4*k+4);

% to take every fourth character and check whether

% it is ref or speaker

    if Y == 'MSDC'

% ************Change to speaker name***********

        speaker(j)=B(i);

% Put the speaker prob values into array speaker.

        j=j+1;

    else

        ref(m)=B(i);

% Put the prob values of ref into array ref

        m=m+1;
    end
    k=k+1;
end
Diff= speaker-ref;
% to get positive values in case the log prob values of
% speaker is better than reference.
for n=1:length(Diff)
    sum=sum+Diff(n);
end
mean=sum/length(Diff);
for p=1:length(Diff)
    var=var + (Diff(p) - mean)^2;
end
stddev=sqrt(var/length(Diff));
for q=1:length(Diff)
    norm(q)=(Diff(q)-mean)/stddev;
end
fclose(fid);
result=[speaker’ ref’ Diff’ norm’];
result1=[speaker; ref; Diff; norm];
% speaker1=num2str(speaker);
% ref1=num2str(ref);
% Diff1=num2str(Diff);
% norm1=num2str(norm);
```

```
fid1=fopen('D:\SV2\results\trainingdata\mdcs2.xls','w');

fprintf(fid1, '%f\t %f\t %f\t %f\n', result1);

fclose(fid1);

clear A B C X Y fid i j k m n p q sum var fid1 norm1;
```

SOURCE CODE FOR SPEAKER RECOGNITION FROM ENCRYPTED VOIP COMMUNICATIONS

C.1 Matlab Code for Extracting Packet Lengths

```
%This file extracts the segments length from OGG encoded files
fid=fopen('D:/OGIVOIP/ogg/0042/00421aa1', 'r');
 B=fread(fid);
 fclose (fid);
 num_pages=0;
 start=0;
 for i=1:length(B)-1
     if B(i)==79 && B(i+1)==103 && B(i+2)==103 && B(i+3)==83
        num_pages=num_pages+1; start_page(num_pages)=i;
         if num_pages > 2
               num_segment_index=i+26;
             for j=1:B(num_segment_index)
                  segment_length(start+j)=B(num_segment_index+j);
            end
             start=start+B(num_segment_index);
         end
```

```
    end
end
C=segment_length;
fid1=fopen('D:/OGIVOIP/SegmentLengths/0042/00421aa1', 'a');
fwrite(fid1,C);
fclose(fid1);
clear segment_length C A fid fid1 start B i j k m
num_segment_index num_pages start_page;
```

C.2 Matlab Code for Calculating Trigram Probabilities of Packet Lengths

```
%This file calculates the Trigram Probabilities
%of the Packet Lengths
fid=fopen('D:/OGIVOIP/SegmentIndex/1022/10221aa1','r');
C=fread(fid);
fclose(fid);
count=zeros(8,8,8);
m=1;
for i=1:8
    for j=1:8
        for k=1:8
            for m=1:length(C)-2
                if C(m)==i && C(m+1)==j && C(m+2)==k
                    count(i,j,k)=count(i,j,k)+1;
                end
            end
        end
    end
end
prob=count/(length(C)-2);
prob=reshape(prob,512,1);
fid1=fopen('D:/OGIVOIP/SegmentProb/1022/10221aa1','w');
fwrite(fid1,prob, 'single');
fclose(fid1);
clear fid fid1 m i j k count prob ans C;
```

C.3 MATLAB Code for Counting the Occurrences

```
%This file counts the number of occurrences of a specific
%packet length
fid=fopen('D:/OGIVOIP/ogg/1022/10221aa1', 'r');
B=fread(fid);
fclose (fid);
num_pages=0;
start=0;
for i=1:length(B)-1
    if B(i)==79 && B(i+1)==103 && B(i+2)==103 && B(i+3)==83
      num_pages=num_pages+1; start_page(num_pages)=i;
        if num_pages > 2
            num_segment_index=i+26;
```

```
                for j=1:B(num_segment_index)
                   segment_length(start+j)=B(num_segment_index+j);
               end
               start=start+B(num_segment_index);
           end
       end
   end
   C=segment_length;
   A=[6 10 15 20 28 38 46 62];
   for k=1:8
   for m=1:length(C)
   if C(m)==A(k)
   C(m)=k;
   end
   end
   end
   fid1=fopen('D:/OGIVOIP/SegmentIndex/1022/10221aa1', 'a');
   fwrite(fid1,C);
   fclose(fid1);
   clear segment_length C A fid fid1 start B i j k m
   num_segment_index num_pages start_page;
```

BIBLIOGRAPHY

[1] D. Burke, *Speech Processing for IP Networks: Media Resource Control Protocol (MRCP)* (John Wiley & Sons, 2007).

[2] L. Rabiner and R. Schafer, *Digital Processing of Speech Signals* (Pearson Education, 2004).

[3] ITU-T Recommendations G.711, “Pulse Code Modulation (PCM) of Voice Frequencies,” Technical report (Nov, 1988) .

[4] S. Pramanik, ”Finding a Way to Break Speech Cipher”, Master’s thesis, Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India, 2002.

[5] D. R. Root, “FNBT Secure Telephone Units:Considerations for Implementation into Tactical Communication Networks,”, A White Paper by Ultra Electronics DNE Technologies, 2005.

[6] Y. Gao, E. Shlomot, A. Benyassine, J. Thyssen, H. yu Su, and C. Murgia, “The SMV algorithm selected by TIA and 3GPP2 for CDMA applications,” In *ICASSP ’01: Proceedings of the Acoustics, Speech, and Signal Processing, 200. on IEEE International Conference*, pp. 709–712 (IEEE Computer Society, Washington, DC, USA, 2001).

[7] I. D.Curcio, J. Kalliokulju, and M. Lundan, “AMR Mode Selection Enhancement in 3G Networks,” Multimedia Tools Appliations **28,** 259–281 (2006).

[8] http://www.itu.int/rec/T-REC-G/e.

[9] http://ilbcfreeware.org/.

[10] L. Rabiner and B. H. Juang, *Fundamentals of speech recognition* (Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993).

[11] A. G. Warner, R. D. Hughes, and R. A. King, “A Direct Voice Input Man-Machine Interface Strategy to Provide Voice Access for Severely Impaired Speakers,” In *Proceedings of the UK IT conference*, pp. 279–285 (IEEE Computer Society, Southampton, UK, 1990).

[12] J. Keshet and S. Bengio, *Automatic Speech and Speaker Recognition:Large Margin and Kernel Methods* (John Wiley & Sons Inc., 350 Main Street Malden MA 02148 USA, 2009).

[13] B. G. Lee and S. C. Kim, *Scrambling Techniques For Digital Transmission* (Springer Verlag, 1995).

[14] A. J. Menezes, P. C. Orschot, and S. Vanstone, *Handbook of Applied Cryptography* (CRC Press, 1997).

[15] L. A. Khan, M. Baig, and M. Ashraf, “Exploiting Silence for Ciphertext Only reconstruction of Stream Ciphered Digitized Voice,” In *Proceedings of the 2007 International Conference on Security and Management (SAM07),*, (June, 2007).

[16] L. A. Khan and M. Baig, “Cryptanalysis of Keystream Reuse in Stream Ciphered Digitized Speech using HMM based ASR Techniques,” In *Proceedings of the International Conference on Electrical Engineering*, (Oct, 2007).

[17] L. A. Khan and M. S. Baig, “Automated Cryptanalysis of Plaintext XORs of Waveform Encoded Speech,” IAENG International Journal of Computer Science **35,** 234–241 (May, 2008).

[18] L. A. Khan and M. Baig, in *Advances in Computational Algorithms and Data* (Springer-

Verlag, 2008), Chap. A Hidden Markov Model Based Speech Recognition Approach to Automated Cryptanalysis of Two Time Pads.

[19] L. A. Khan and M. S. Baig, “Cryptanalysis of Two Time Pads in case of Compressed Speech,” Submitted to Computers and Electrical Engineering (2009).

[20] L. A. Khan and M. Baig, “A Server Based ASR Approach to Automated Cryptanalysis of Two Time Pads in case of Speech,” In *Proceedings of the International Conference on Information Assurance and Security (IAS08)*, (Sep, 2008).

[21] L. A. Khan, F. Iqbal, and M. S. Baig, “Speaker Verification from Partially Encrypted Compressed Speech,” Accepted in Digital Investigation (2010).

[22] L. A. Khan, M. S. Baig, and A. M. Youssef, “Speaker Recognition from Encrypted VoIP Communications,” Accepted in Digital Investigation (2009).

[23] ITU-T Recommnedations G.763, “Digital Circuit Multiplication Equipment using ADPCM and Digital Signal Interpolation,” Technical report (1998) .

[24] P. Chandra Sehkar, ”Analyzing Encrypted Speech”, Master’s thesis, Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India., 2001.

[25] T. Siegenthaler, “Decrypting a Class of Stream Ciphers Using Ciphertext Only,” IEEE Transactions on Computers **C-34,** 81–85 (1985).

[26] A. Canteaut and E. Filiol, “Ciphertext Only Reconstruction of Stream Ciphers based on Combination Generators,” In *FSE 2000, LNCS 1978*, (Springer Verlag, 2000).

[27] E. Filiol, “Decimation Attack of Stream Ciphers,”, Cryptology ePrint Archive, Report 2000/040, 2000, http://eprint.iacr.org/.

[28] H. Wu and B. Preneel, “Distinguishing Attack on Stream Cipher Yamb,” In *eSTREAM, ECRYPT Stream Cipher Project, Report 2005/043*, (2005).

[29] J. Y. Cho and J. Pieprzyk, “Linear Distinguishing attack on NLS,” In *State of the Art Stream Ciphers Workshop (SASC)*, (2006).

[30] G. Rose and P. Hawkes, “On the applicability of Distinguishing attacks Against Stream Ciphers,” In *http://eprint.iacr.org/2002/142.pdf*,

[31] E. S. Selmer, ”Linear Recurrence Relations over Finite Fields”, Ph.D. thesis, University of Bergen, Norway, 1996.

[32] A. Canteaut and M. Trabbia, “Improved Fast Correlation Attacks Using Parity Check Equations of Weight 4 and 5,” In *Advances in Cryptology EUROCRYPT 2000, LNCS 1807*, pp. 573–588 (Springer Verlag, 2000).

[33] N. Maneli, F. Faramarz, and O. Tatsuaki, “A Fast Correlation Attack via Unequal Error Correcting LDPC Codes,” In *CT-RSA 2004, LNCS vol. 2964*, pp. 54–66 (2004).

[34] A. A. Huurdeman, *The Worldwide History of Telecommunications* (John Wiley and Sons/IEEE Press, 2003).

[35] T. Siegenthaler, “Correlation Immunity of Non Linear Combining Functions for Cryptographic Applications,” IEEE Transactions on Information Theory **35 No.35,** 776–780 (1984).

[36] C. Shannon, “A Mathematical Theory of Communication,” Bell System Technical Journal **27,** 379–423 (1948).

[37] J. Mason, K. Watkins, J. Eisner, and A. Stubblefield, "A Natural Language Approach to Automated Cryptanalysis of Two Time Pads," In *In 13th ACM Conference on Computer and Communications Security*, (Nov, 2006).

[38] H. Wu, "The Misuse of RC4 in Microsoft Word and Excel," Cryptology ePrint Archive, Report 2005/007 (2005).

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

[40] T. Kohno, "Attacking and Repairing the WinZip Encryption Scheme," In *In 11th ACM Conference on computer and communications security*, pp. 72–81 (Oct, 2004).

[41] B. Schneier, Mudge, and D. Wagner, "Cryptanalysis of Microsoft PPTP Authentication Extensions (ms-chapv2)," In *CQRE99*, (1999).

[42] M. Dworkin, "Recommemdations for Block Cipher Modes of Operations," in *NIST Special publication 800-38A, 2001*

[43] D. A. Mcgrew and J. Viega, "The GaloisCounter Mode of Operation (GCM)," Technical report, NIST (May,2005) .

[44] R. Houseley and A.Corry, "GigaBeam High Speed Radio Link Encryption, RFC 4705," Technical report (October, 2006) .

[45] B. Schneier, *Applied Cryptography, Protocols, Algorithms and Source Code in C (Second Edition)* (John Wiley and Sons, 2002).

[46] R. Rubin, "Computer Methods for Decrypting Random Stream Ciphers," Cryptologia 2(3) pp. 215–231 (July,1978).

[47] E. Dawson and L. Nielsen, “Automated Cryptanalysis of XOR Plaintext Strings,” Cryptologia 20(2) pp. 165–181 (April, 1996).

[48] E. Dawson, “Design of a Discrete Cosine Transform based Speech Scrambler,” Electronics Letters **27,** 613–614 (Mar, 1991).

[49] C. P. Wu and C. J. Kuo, “Fast Encrytion Methods for Audio Visual Data Confidentiality,” In *Proccedings of the SPIE, vol 4209*, pp. 284–295 (Nov, 2000).

[50] D. Pearce, “Enabling New Speech Driven Services for Mobile Devices: An Overview of the ETSI standards Activities for Distributed Speech Recognition Front Ends.,” In *AVIOS 2000: The Speech Applications Conference*, (May, 2000).

[51] M. J. Gales, B. Jia, X. Liu, K. C. Sim, P. C. Woodland, and K. Yu, In ,

[52] M. R. Schroeder and B. S. Atal, “Code-excited Linear Prediction (CELP): High-Quality Speech at Very Low Bit Rates,” In *Proceedings of ICASSP, vol. 10*, pp. 937–940 (1985).

[53] M. H. Johnson and A. Alwan, in *Wiley Encyclopedia of Telecommunications*, J. G.Proakis, ed., (John Wiley and Sons Inc., 2003), Chap. Spech Coding: Fundamentals and Applications.

[54] R. Goldberg and L. Riek, *A Practical Handbook of Speech Coders* (CRC Press, NY, 2000).

[55] S. F. Chen and J. Goodman, “An Empirical Study of Smoothing Techniques for Language Modeling,” In *In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 310–318 (1996).

[56] P. F. Brown, V. J. Pietra, P. V. DeSouza, J. C. Lai, and R. L. Mercer, “Class-based n-gram Models of Natural Language,” Computational Linguistics **18,** 467–479 (December, 1992).

[57] L. R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” Proceedings of the IEEE **77,** 257–286 (Feb, 1989).

[58] L. Rabiner and B. H. Juang, *Fundamentals of speech recognition* (Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993).

[59] C. S. Myers and L. R. Rabiner, “A Comparative Study of Several Dynamic Time-Warping Algorithms for Connected Word Recognition,” The Bell System Technical Journal **60,** 1389–1409 (September 1981).

[60] P. Wright, *Spy Catcher* (Viking, New York, NY, 1987).

[61] R. L. Benson and M. Warner, “VENONA: Soviet Espionage and the American Response 1939-1957,” Central Intelligence Agency (1996).

[62] B. Goldburg, E. Dawson, and S. Sridharan, “The Automated Cryptanalysis of Analog Speech Scramblers,” In *EUROCRYPT91, Springer-Verlag LNCS 457*, p. 422 (April, 1991).

[63] R. V. Cox, N. S. Jayant, and B. J. McDermott, “An Analog Scrambler for Speech Based on Sequential Permutations in Time and Frequency,” In *Globecom ’82 - Global Telecommunications Conference, Miami, FL, November 29-December 2, 1982, Conference Record. Volume 1 (A84-26401 11-32). New York, Institute of Electrical and Electronics Engineers, 1982, p. 168-172.*, **1,** 168–172 (1982).

[64] B. Goldburg, E. Dawson, and S. Sridharan, “A Secure Analog Speech Scrambler Using the Discrete Cosine Transform,” In *ASIACRYPT ’91: Proceedings of the International Conference on the Theory and Applications of Cryptology*, pp. 299–311 (Springer-Verlag, London, UK, 1993).

[65] W. W. Chang, J. Y. Wang, and T. H. Tan, "Statistical Cryptanalysis of DFT-based Speech Scramblers," In *Proceedings of ISITA '94: International Symposium on Information Theory and Its Applications*, pp. 1191–1195 (National conference publication (Institution of Engineers, Australia), 1994).

[66] S. Li, C. Li, G. Chen, N. G. Bourbakis, and K. T. Lo, "A General Quantitative Cryptanalysis of Permutation-only Multimedia Ciphers Against Plaintext Attacks," Image Communications **23,** 212–223 (2008).

[67] I. J. Kumar, *Speech Cryptanalysis in Cryptology: System Identification and Key Clustering* (SB, Aegean Park Press, 1997).

[68] A. Narayan and V. Shmatikov, "Fast Dictionary Attacks on Human-Memorable Passwords using Time-Space Trade-Off," In *12th ACM Conference on Computer and Communications Security*, pp. 364–372 (Washington D.C., Nov, 2005).

[69] D. X. Song, D. Wagner, and X. Tian, "Timing Analysis of Keystrokes and Timing Attack on SSH," In *10th USENIX Security Symposium*, (Aug, 2001).

[70] 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).

[71] D. Asonov and R. Agrawal, "Keyboard Acoustic Emanations," In *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 3–11 (2004).

[72] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard Acoustic Emanations Revisited," In *12th ACM Conference on Computer and Communications Security*, pp. 373–382 (Nov, 2005).

[73] C. Karlof and D. Wagner, “Hidden Markov Models Cryptanalysis,” In *Cryptographic Hardware and Embedded Systems- CHES03, LNCS 2779*, pp. 17–34 (Springer-Verlag, 2003).

[74] *BEEP-British English Pronounciation Dictionary (Phonetic Transcriptions of over 250,000 English words)* (Springer Verlag, 1992).

[75] S. J. Young, G. Evermann, T. Hain, D. Kershaw, G. L. Moore, J. J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book* (Cambridge University, Cambridge, 2003).

[76] V. Tyagi and C. Wellekens, “On Desensitizing the Mel-Cepstrum to Spurious Spectral Components for Robust Speech Recognition,” In *Proceedings of ICASSP, vol. 1*, p. 529532 (2005).

[77] J. J. Godfrey, E. C. Holliman, and J. McDaniel, “SWITCHBOARD: Telephone Speech Corpus for Research and Development,” In *Proceedings of ICASSP*, p. 234 (1992).

[78] A. Servetti and J. C. Martin, “Perception-Based Partial Encryption of Compressed Speech,” IEEE Transactions on Speech and Audio Processing **10,** 637–643 (Nov, 2002).

[79] J. P. Campbell, T. E. Tremain, and V. C. Welch., “The Federal Standard 1016 4800 bps CELP Voice Coder,” Digital Signal Processing **1,** 145–155 (1991).

[80] ITU-T Recommendations G.723.1, “The Federal Standard 1016 4800 bps CELP Voice Coder,” Dual Rate Speech Coders for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s, 1996. .

[81] 3GPP TS 26.290, “Audio Codec Processing Functions; Extended Adaptive Multi-Rate

Wideband (AMR-WB+) codec; Transcoding functions, version 6.3.0 (2005-06)," Technical report, 3rd Generation Partnership Project (3GPP) (2005) .

[82] ITU-T Recommendations G.728, "Coding of Speech at 16 kbits/s using Low Delay Code Excited Linear Prediction," Technical report (Sep, 1992) .

[83] ITU-T Recommendations G.729, "Coding of Speech at 8 kbits/s Conjugate Structure Algebraic Code Excited Linear Prediction," Technical report (1996) .

[84] T. E. Tremain, "The Government Standard Linear Predictive Coding Algorithm: LPC-10," Speech Technology Magazine pp. 40–49 (April 1982).

[85] "ETS 300 961: Digital cellular telecommunications system (Phase 2+); Full rate speech; Transcoding (GSM 06.10 version 5.1.1)," (May, 1998).

[86] L. A. Khan and M. S. Baig, "Cryptanalysis of Keystream Reuse in Stream Ciphered Digitized Speech using HMM based ASR Techniques," In ,

[87] D. Zaykovskiy, "Survey of The Speech Recognition Techniques For Mobile Devices," A TechRepublic White Paper (Jun, 2007).

[88] "Recognition Performance Evaluations Of Codecs For Speech Enabled Services (SES)," Technical report, 3GPP TR 26.943 (Dec, 2004) .

[89] J. M. Heurta, "Speech Recognition in Mobile Environments", Ph.D. thesis, Carnegie Mellon University, April, 2000.

[90] B. Raj, J. Migdal, and R. Singh, "Distributed Speech Recognition with Codec Parameters," In *In proceedings ASRU2001*, (Dec, 2001).

[91] C. Pelaez, A. Gallardo-Antolin, and F. D. de Maria, "ecognizing Voice over IP: A Robust Front-End For Speech Recognition on The World Wide Web," IEEE Tran. on Multimedia 3 (2001).

[92] F. Bimbot, J. F. Bonastre, C. Fredouille, G. Gravier, I. M. Chagnolleau, S. Megnier, T. Merlin, J. O. Garcia, D. P. Delacretaz, and D. A. Reynolds, "A Tutorial on Text-Independent Speaker Verification.," EURASIP Journal on Applied Signal Processing **4,** 430–451 (2004).

[93] A. Braun and H. Kunzel, "Is Forensic Speaker Identification Unethical -Or can it be Unethical Not to do it?," In *Proceedings of RLA2C Workshop on Speaker Recognition and its Commercial and Forensic Applications*, pp. 145–148 (Avignon, France, 1998).

[94] J. Koolwaij and L. Boves, "On the Use of Automatic Speaker Verification Systems in Forensic Casework," Audio and Video- based Biometric Person Authentication (1999).

[95] C. V. Wright, L. Ballard, S. E. Coull, F. Monrose, and G. M. Masson, "Spot Me if You Can: Uncovering Spoken Phrases in Encrypted VoIP Conversations," In *SP '08: Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pp. 35–49 (2008).

[96] J. Goodman and A. P. Chandrakasan, "Low power scalable encryption for wireless systems," Wireless Netwroks 4 (1998).

[97] T. Lookabaugh, I. Vedual, and D. C. Sicker, "Selective Encryption and MPEG-2," ACM Multimedia (2003).

[98] "Coding of Speech at 8kbits/s Conjugate Structure Algebraic Code Exited Linear Prediction,", 1996.

[99] M. Matsui, "Linear cryptanalysis method for DES Cipher," In Advances in Cryptology - EUROCRYPT'93, LNCS 765 (1994).

[100] D. A. Reynolds and R. C. Rose, “Robust Text Independent Speaker Identification Using Gaussian Mixture Speaker Models,” IEEE transactions of speech and audio processing 3 (1995).

[101] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Robust Text Independent Speaker Identification Using Gaussian Mixture Speaker Models,” Digital Signal Processing **10,** 19–41 (2000).

[102] F. Bimbot, J. F. Bonastre, C. Fredouille, G. Gravier, I. M. Chagnolleau, S. Megnier, T. Merlin, J. O. Garcia, D. P. Delacretaz, and D. A. Reynolds, “A Tutorial on Text-Independent Speaker Verification.,” EURASIP Journal on Applied Signal Processing **4,** 430–451 (2004).

[103] T. Kinnun, J. Saastamoinen, V. Hautamaki, M. Vinni, and P. Franti, “Comparing Maximum a Posteriori Vector Quantization and Gaussian Mixture Models in Speaker Verification.,” In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 145–148 (Taiwan, 2009).

[104] M. Gales and S. Young, “The Applications of Hidden Markov Models in Speech Recognition,” Foundations and Trends in Signal Processing **1,** 195–304 (2008).

[105] T. Kinnun, J. Saastamoinen, V. Hautamaki, M. Vinni, and P. Franti, “Comparing Maximum a Posteriori Vector Quantization and Gaussian Mixture Models in Speaker Verification.,” In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, (2009).

[106] A. Acero, ”Acoustical and Environmental Robustness in Automatic Speech Recognition”, Ph.D. thesis, Department of Elect and Comp Engg, Carnegie Mellon University, Pittsburgh, 1990.

[107] http://www.voiceage.com/openinit_g729.php.

[108] N. Provos, "Voice Over Misconfigured Internet Telephones (VOMIT). http://vomit.xtdnet.nl/,".

[109] L. G. Kersta, "Voiceprint Identification," Nature **196,** 1253–1257 (1962).

[110] D. A. Reynolds, "An Overview of Automatic Speaker Recognition Technology," In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, (2002).

[111] A. Martin and M. Przybocki, "The NIST Speaker Recognition Evaluation Series,", National Institute of Standards and Technology Web site, 2009.

[112] M. A. Przybocki, A. F. Martin, and A. N. Le, "NIST Speaker Recognition Evaluation Chronicles, Part 2," In *IEEE Odysee, ISCA Speaker Recognition Workshop*, (2006).

[113] M. A. Przybocki, A. F. Martin, and A. N. Le, "NIST Speaker Recognition Evaluations Utilizing the Mixer Corpora-2004, 2005, 2006," IEEE Transactions on Audio Speech and Language Processing **15,** 1951–1959 (2007).

[114] Advisory Panel on White House Tapes, "The Executive Office Building Tape of June 20, 1972: Report on a Technical Investigation," Technical report, United States District Court for the District of Columbia (1974) .

[115] J. S. Sachs, "Graphing the Voice of Terror," Popular Science pp. 38–43 (2003).

[116] J. Koolwaaij and L. Boves, "On the Use of Automatic Speaker Verification Systems in Forensic Casework," In *Proceedings of Audio and Video- based Biometric Person Authentication*, pp. 224–229 (1999).

[117] M. Baugher and D. McGrew, “RFC 3711: SRTP: The Secure Real Time Transport Protocol,”, IETF, 2003.

[118] W. Campbell, D. Sturim, D. Reynolds, and A. Solomonoff, “SVM-based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation,” In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 1–97 (2006).

[119] C. V. Wright, L. Ballard, F. Monrose, and G. M. Masson, “Language Identification of Encrypted VoIP Traffic: Alejandra y Roberto or Alice and Bob?,” In *SS’07: Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, (2007).

[120] W. C. Chu, *Speech Coding Algorithms* (John Wiley and Sons, 2003).

[121] A. Acero, “Acoustical and Environmental Robustness in Automatic Speech Recognition,” Foundations and Trends in Signal Processing **1,** 195–304 (1993).

[122] J. M. Valin and C. Montgomery, “Improved Noise Weighting in CELP Coding of Speech-Applying the Vorbis Psychoacoustic Model to Speex,” In *Audio Engineering Society Convention*, (2006).

[123] R. Barbeieri, D. Bruschi, and E. Rosti, “Voice over IPSec: Analysis and Solutions,” In *Proceedings of the 18 Annual Computer Security Applications Conference*, pp. 261–270 (2002).

[124] S. Pfeiffer, “RFC 3533: The Ogg Encapsulation Format Version 0,”, IETF, 2003.

[125] R. Koenen, “ISO/IEC JTC1/SC29/WG11:Coding of Moving Pictures and Audio,”, 2002.

[126] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occuring in the Statistical Analysis of Probabalistic Fuctions of Markov Chains," Annals of Mathematics and Statistics 4 (1970).

[127] G. D. Forney, "The Viterbi Algorithm," Proceedings of the IEEE 61 (1973).

[128] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," Journal of Royal Statistical Society 39 (1977).

[129] G. McLachlan, *Mixture Models* (Marcel Decker, New York, 1988).

[130] E. Frank and S. Kramer, "Ensembles of Nested Dichotomies for Multi-class Problems," In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, (2004).

[131] J. R. Quinlan, *C4.5: Programs for Machine Learning* (Morgan Kaufmann, 1992).

[132] A. Agresti, *Categorical Data Analysis* (John Wiley and Sons, 2002).

[133] A. J. Smola, B. Schlkopf, and B. S. Olkopf, "A Tutorial on Support Vector Regression," Technical report, Statistics and Computing (2003) .

[134] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery **2,** 121–167 (1998).

[135] C. W. Hsu, C. C. Chang, and C. J. Lin, "A Practical Guide to Support Vector Classification," Technical report, Department of Computer Science National Taiwan University, Taipei, Taiwan (2009) .

[136] R. Cole, M. Noel, and V. Noel, "The CSLU Speaker Recognition Corpus," In *Proceedings*

of the International Conference on Spoken Language Processing, pp. 3167–3170 (Australia, 1998).

[137] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. (Margan Kaufmann, San Francisco, 2005).

[138] D. Heckerman, "A Tutorial on Learning Bayesian Networks," Technical report, Microsoft Research Technical Report (1995) .

[139] F. Yoav and R. E. Schapire, "Experiments with a New Boosting Algorithm," In *International Conference on Machine Learning*, pp. 148–156 (1996).

[140] J. Su, H. Zhang, C. X. Ling, and S. Matwin, "Discriminative Parameter Learning for Bayesian Networks," In *International Conference on Machine Learning*, pp. 1016–1023 (2008).

[141] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifiers," Machine Learning 29 (1977).

[142] J. C. Platt, "Fast Training of Support Vector Machines Using Sequential Minimal Optimization," pp. 185–208 (1999).