Wiretap-proof: What They Hear is Not What You Speak, and What You Speak They Do Not Hear

Hemant Sengar
Technology Development
VoDaSec Solutions
Fairfax, VA
hsengar09@gmail.com

Haining Wang
Dept. of Computer Science
College of William and Mary
Williamsburg, VA
hnw@cs.wm.edu

Seyed Amir Iranmanesh
Dept. of Computer Science
College of William and Mary
Williamsburg, VA
sairan@cs.wm.edu

ABSTRACT

It has long been believed that once the voice media between caller and callee is captured or sniffed from the wire, either legally by law enforcement agencies or illegally by hackers through eavesdropping on communication channels, it is easy to listen into their conversation. In this paper, we show that this common perception is not always true. Our real-world experiments demonstrate that it is feasible to create a hidden telephonic conversation within an explicit telephone call. In particular, we propose a real-time *covert* communication channel within two-way media streams established between caller and callee. The real-time covert channel is created over the media stream that may possibly be monitored by eavesdroppers. However, the properly encoded media stream acts as a cover (or decoy) carrying bogus media such as an earlier recorded voice conversation. This spurious content will be heard if the media stream is intercepted and properly decoded. However, the calling and called parties protected by the covert communication channel can still directly talk to each other in privacy and real-time, just like any other normal phone calls. This work provides an additional security layer against media interception attacks, however it also exposes a serious security concern to CALEA (Communications Assistance for Law Enforcement Act) wiretapping and its infrastructure.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*

Keywords

Wiretapping, Media Eavesdropping, Covert Communication

1. INTRODUCTION

There are two kinds of eavesdroppers attempting to intercept voice media and listen into telephonic conversation. The first group belongs to illegal eavesdroppers who observe traffic (signaling, media or both) and try to learn who is calling whom and possibly the content of their communication. There are many examples of illegal interception of voice media. In January 2012, a trans-Atlantic call between the FBI and the UK's Scotland Yard in which operatives from the two law enforcement agencies discussed ongoing cases regarding a number of alleged hackers was intercepted, recorded by hackers and later uploaded on the web [20]. A few years ago, there was a Greek wiretapping case involving the illegal tapping of more than 100 mobile phones on the Vodafone Greece network, most of which belong to members of the Greek government and top-ranking civil servants [26]. In a corporate world, we can find many examples of illegal eavesdropping on CEOs or some other targets' phone calls attempting to learn about corporate strategies or financial information.

The second group of eavesdroppers belong to lawfully authorized surveillance, in which a target interception must be approved from the courts or a law enforcement agency. For example, there are recent examples of Illinois governor Rod Blagojevich's attempt of selling senate seat [29] and federal prosecutions of Raj Rajaratnam insider-trading case [28] using wiretapped telephone calls as crucial evidence. Lawfully authorized electronic surveillance is a critical tool used by law enforcement for investigative purposes and also for the prosecution of serious crimes. Most of the democratic countries in the world have electronic surveillance infrastructure and its associated rules and regulations in place. For example, the 1994 USA law for Communications Assistance for Law Enforcement Act (CALEA) requires telecommunication service providers to incorporate various capabilities for law enforcement wiretapping into their networks. These capabilities have been in place for many years in circuit-switched voice networks, i.e., public switched telephone networks (PSTN), to intercept and identify calling and called party information along with the communication content. Moreover, it is also required by the law that VoIP service providers must also be CALEA compliant and provide standard interfaces to their equipments for capturing call-related meta data (who is calling whom) and media content.

The focus of this paper lies in how to protect important information from falling into eavesdroppers' hands. In general, there are two methods – either make it indecipherable or hide in plain sight. The first method is known as encryption. The encrypted messages are secure against general prying eyes. However, plainly visible encrypted messages – no matter how unbreakable – will arouse suspicion, and may in themselves be incriminating in countries where encryption is illegal [30]. The second method is known as steganography, where a secret message is embedded within another cover message in such a way that an observer is not aware of anything unusual and does not have any suspicion. As today's computer and network technologies provide ready-made communication channels for steganography, it is believed that steganogra-

CODASPY'14, March 3–5, 2014, San Antonio, Texas, USA.
Copyright 2014 ACM 978-1-4503-2278-2/14/03 ...$15.00.
.

phy has become a favorable communication channel for terrorists to conduct their activities [6, 10]. To date, steganography is used to hide bits and pieces of information by modifying cover medium's redundant bits. There are many commercial and open source software that can hide information in various types of digital media, such as images, audio, and text files, generally using their least significant bits (LSBs).

1.1 Existing Audio Steganography Methods

In audio steganography, tools such as `S-Tools` [2], `MP3Stego` [8], and `Hide4PGP` [11] employ standard embedding method of using LSB with `WAV`, `MP3`, and `VOC` audio file as cover media, respectively. The `SteganRTP` [16] tool also uses LSB, but it utilizes real-time media sessions as cover medium. For further improvement, Takahashi et al. [21] placed CELP-based codec (i.e., `G.729`) audio data within LSBs of `G.711` generated audio. Similarly, Wang et al. [23] used Speex codec to hide compressed audio within LSBs of `G.711` audio packets. As we note, there is a common thread running across all these audio steganography tools. This common thread is a well-known approach of using LSBs of cover media, because of its high capacity or throughput. Consequently, many methods and tools exist today to detect LSB-based covert channels [7, 19, 9, 24]. The other known audio steganography methods, such as spread spectrum, phase coding, and echo data hiding [14], are not very relevant to our work, as they cannot provide channel capacity high enough to hide real-time voice communication.

1.2 Challenges

To the best of our knowledge, except using LSBs, we are not aware of any other efforts of using audio steganography techniques in real-time communication channels to hide a real-time voice communication. It is mainly because of two reasons: (1) voice is time sensitive media and (2) its presentation requires at least several thousands bits of information per second. However, if we could show that it is possible to hide a telephonic conversation by creating a covert communication channel within another (i.e., cover) conversation, without using LSBs or any other previously known audio steganography methods, it will have profound effects on call monitoring and media interception. On one hand, it will provide a new security approach against illegal eavesdroppers; on the other hand, it will induce a serious security implication to CALEA and its infrastructure.

With the wide use of VoIP within enterprise networks, it is speculated that the confidential data (audio, image, text etc.) can be embedded and transmitted out of the networks via RTP streams. However, since LSB-based covert channels are easily detectable and have poor immunity to manipulation, the LSB-based methods have never been a serious threat. For example, it is well-known fact that such covert channels can be easily removed either by randomizing the LSBs or passing the audio stream through a transcoding process (e.g., converting 64 Kbps `G.711` audio channel to 8 Kbps `G.729` audio channel). None of the existing audio covert channels can survive after the transcoding process. In this paper, we attempt to develop a new method and experimentally demonstrate that even after transcoding, it is still feasible to recover and reconstruct the *lost* or *obfuscated* covert channel.

1.3 Contributions

In this paper, through real-world experiments, we demonstrate that it is feasible to create a real-time covert voice communication channel within an explicit and open media channel. We propose a new audio steganography method that is unique in several aspects: (1) Waveform codec approach – in all previous approaches, compressed codecs such as `G.729` (8Kbps) and `Speex` (2-44 Kbps) are used to hide audio within `G.711` codec (64 Kbps) audio due mainly to low bandwidth requirement of compressed codecs (i.e., covert channel capacity is always lower than cover channel capacity). In our approach, we take a radically different approach by hiding 64 Kbps worth of information within another 64 Kbps `G.711` encoded cover audio. (2) Cover audio sample replacements – in our approach, the cover and covert audio samples are interleaved with each other, a few of the cover audio samples are replaced with the covert audio samples, instead of modifying the bits of cover samples. (3) Reconstruction of imprecise waveform – at the receiver side, based on the limited number of covert audio samples, we discover all the missing samples and reconstruct a waveform that is approximate to the original one. (4) Transcoding process – as a sample-based approach, we can still recover the covert samples, even though the RTP stream may have undergone through the transcoding process reducing 64 Kbps `G.711` stream to 8 Kbps `G.729` stream. (5) Codebook-based approach – compared to LSB-based approaches, now peers have the flexibility to create their own private communication in many different ways, making the covert channel unpredictable and hard to decipher for eavesdroppers.

1.4 Brief Overview

Let's consider a two-party call where two audio streams between caller and callee undergo the encoding and decoding processes at the sender and receiver sides. At the sender side, the earlier recorded conversation is used as a cover media. Some of its samples are replaced with the samples that have some specific key characteristics of the real-time voice spoken over the microphone, and then the mixed samples are encoded, packetized and transmitted across the networks. At the receiver side, once the samples with the hidden voice characteristics and its time line (i.e., temporal relationship) are separated from the cover, the cover media is discarded. Using the received characteristic samples and its time information, we reconstruct the spoken words or phrases and then send it to the receiver-side speaker to play it out. Even if the properly decoded media is intercepted anywhere between caller and callee, it will still be very hard to guess or reconstruct the hidden communication. The intercepted media will be playing explicit spurious cover content only. Here we merely describe the one-way media stream operation, however, it should be noted that the same process is also be repeated in other direction to establish two-way media streams.

The remainder of the paper is structured as follows. In Section 2, we discuss the background of this work, including SIP-based IP telephony, CALEA, VoIP media stream, and conventional audio steganography methods. In Section 3, we describe our new real time voice steganography technique. In Sections 4 and 5, we present the encoding and decoding processes, respectively. In Section 6, we validate the efficacy of the proposed approach through real experiments. In Section 7, we survey related work. Finally, we conclude the paper in Section 8.

2. BACKGROUND

While the proposed approach of real-time voice steganography is general enough and applicable to both traditional PSTN and emerging VoIP telephony networks, our main focus is on VoIP networks. This is mainly due to two reasons: (1) VoIP networks provide a flexible platform to perform our proof of concept testing, and (2) our telephone sets are SIP-based softclients, in which we implement the modified approach of encoding and decoding of the media streams. However, it should be noted that the same method could also be implemented on smartphones and hardware-based analog phones.

2.1 SIP-based IP Telephony

The Session Initiation Protocol (SIP) [15], belonging to the application layer of the TCP/IP protocol stack, is used to set up, modify, and tear down multimedia sessions including telephone calls between two or more participants. SIP-based telecommunication architectures have two kinds of elements: end devices referred to as user agents (UAs) and SIP servers. Irrespective of being a software or hardware phone, UAs combine two sub-entities: the connection requester referred as the user agent client (UAC) and the connection request receiver referred to as the user agent server (UAS). Consequently, during a SIP session, both UAs switches back and forth between UAC and UAS functionalities. SIP messages consisting of request-response pairs are exchanged for call set up, from six kinds consisting of `INVITE, ACK, BYE, CANCEL, REGISTER`, and `OPTIONS` - each identified by a numeric code according in RFC 3261 [15].

2.2 Communications Assistance for Law Enforcement Act (CALEA)

The Communications Assistance for Law Enforcement Act (CALEA) is a United States wiretapping law passed in 1994 to regulate telecommunication compliance with lawful surveillance of digitally switched telephone networks. The objective of CALEA is to enhance the ability of law enforcement and intelligence agencies to conduct electronic surveillance. This requires that telecommunications carriers and manufacturers of telecommunications equipment modify and design their equipment, facilities, and services to ensure the built-in surveillance capabilities, allowing federal agencies to monitor all telephone, broadband Internet, and VoIP traffic in real-time [25].

The J-Standard (J-STD-025) defines the interfaces between a telecommunication service provider (TSP) and a Law Enforcement Agency (LEA) to assist the LEA in conducting lawfully authorized electronic surveillance. It is developed by a joint effort of Telecommunications Industry Association (TIA), the Alliance for Telecommunications Industry Solutions (ATIS), and various other industry organizations and interest groups. As a product of the traditional circuit-switched wireline and wireless telecommunications industry associations, the J-standard does not specifically address the requirements of other (competing) technologies such as Voice-over-IP (VoIP). However, J-standard serves as a guide to many other industry associations to develop their own specifications meeting their technical requirements.

Now we discuss how a VoIP service provider implements CALEA compliance. Here a VoIP target subscriber may call to another VoIP subscriber hosted by the same service provider or to an external number (call routed through PSTN networks). The session border controller (SBC) is an edge device between VoIP subscribers and the service provider's core network, and is used to exert control over the signaling and the media streams. The warrant for a particular target (i.e., a subscriber to be monitored) is provisioned on the SBC. The SBC uses directory number (DN) to match the target and intercept a call. The SBC provides intercepted call data events and replicated media for matching targets to the delivery function (DF), and then both content and target call data are relayed to the appropriate LEA's collection function (CF).

2.3 VoIP Media Stream

As a telephone subscriber talks over phone, the telephone device is responsible for capturing and transforming audio for transmission. The media capture process consists of capturing an uncompressed frame and transforming into a format suitable for encoder to generate a compressed frame. The compressed frames are packetized to create one or more (i.e., fragmented) RTP packets. The device may also participate in error correction and congestion control by adapting the transmitted media stream in response to feedback received from the other end.

Audio Capture: When a telephone caller speaks over phone, a device known as microphone responds to sound pressure. The microphone produces a time-varying electrical voltage proportional to the increase or decrease in local pressure that constitutes sound. This continuous time-varying voltage is an electric analog of the acoustic signal. The analog audio signals captured from the microphone are sampled, digitized and stored in a buffer. Once a fixed number of samples have been collected (i.e., a frame is formed), the buffer is made available to the application. The frame is not available to the application until the last sample is collected in the buffer. To avoid delays to the application, the buffer size is close to the frame duration.

Compression: The uncompressed audio data captured in the buffer is passed to encoder to produce compressed frames. Frames can be encoded in several ways depending on the compression algorithm used. Based on the negotiated codec choice between peers, state may be maintained between frames and is made available to the encoder along with each new frame of data. Some of the codecs produce fixed-size frames and some produce variable-size frames as their output.

RTP Packets: Now the frames are ready to be packetized as RTP packets before being transmitted over the network toward the other end, i.e., callee. The RTP packetization routine creates one or more RTP packets for a frame depending upon the maximum transmission unit (MTU) of the network. The packet header and payload are defined according to the used codec specification.

3. REAL-TIME VOICE STEGANOGRAPHY

From voice steganography perspective, it is a challenging task to hide a real-time voice communication within another real-time voice channel. The reason lies in three aspects. (1) *Capacity limitation*, the amount of information can be hidden in cover media is limited, but the voice channel requires at least several thousands of bits of information per second. Only the LSB-based approach is known to have the channel capacity of few thousands bits per second and therefore it is the most dominant method to hide various types of information, such as images, audio, and text. Except the LSB-based method, there is no other method that can achieve the high channel capacity or throughput. (2) *Time domain*, the hidden information has to be related with the time information, because at the receiver end we need to know what information has to be presented at what time. And finally, (3) *real-time presentation*, there are strict timing deadlines that must be met in terms presenting the data. The decoded hidden information must be presented within less than 150 ms at the other end[1]. Now we discuss how we solve these three limitations in our proposed approach while making the resulted covert channel harder to be detected.

3.1 Pulse Code Modulation

In telephony world the most commonly used voice codec is G.711. As a waveform codec, G.711 is an ITU-T standard for audio companding. Its formal name is Pulse code modulation (PCM) of voice frequencies. Non-uniform (logarithmic) quantization with 8 bits is

[1] The user perception of the voice quality starts deteriorating as the one-way latency exceeds 150 ms.

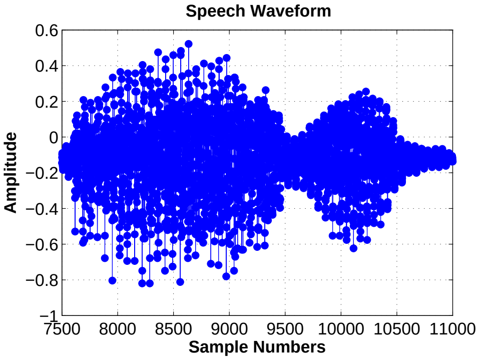

Figure 1: A Snapshot of Voice Utterance [8K Samples per Sec.]

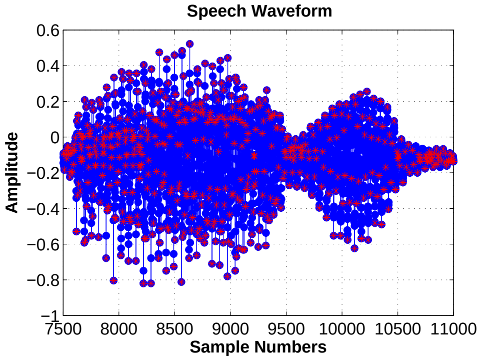

Figure 2: Local Maxima and Minima Data Points (Shown as Red $*$)

used to represent each sample, resulting in a bit rate of 64 kbit/s. There are two slightly different versions: $\mu - law$, which is used primarily in North America, and $A - law$, which is in use in most other countries outside North America.

The Pulse Code Modulation is based on the *Nyquist Frequency*. According to Nyquist Frequency, in order to recover all the Fourier components of a periodic waveform, it is necessary to use a sampling rate at least twice the highest waveform frequency. More formally:

- $x(t)$ is a band-limited signal with bandwidth f_h,
- $p(t)$ is a sampling signal consisting of pulses at internals $T_s = \frac{1}{f_s}$, where f_s is the sampling frequency,
- $x_s(t) = x(t)p(t)$ is the sampled signal,

then $x(t)$ can be recovered exactly from $x_s(t)$ if and only if $f_s \geq 2f_h$. If voice data is limited to frequencies below $4,000$ Hz ,then $8,000$ samples per second would be sufficient to completely characterize the voice signal. Based on this sampling theorem, speaker's utterances captured by the microphone are sampled as shown in Figure 1.

3.2 Capacity Limitation

The biggest challenge in audio steganography is to find a method that is not based on LSBs, but still can have channel capacity high enough to hide information worth of few thousands of bits per second. We know that according to Nyquist theorem $8,000$ samples per second would be sufficient to completely characterize the voice signal if its frequencies are limited below $4K$ Hz. However, the main question is *why do we need exact representation of the speech waveform, when an approximate representation is good enough to human ears?* Since we cannot carry $8,000$ samples worth of information in cover media, our goal is to find a method that can allow us to reconstruct a waveform similar to the original one using as few samples as possible, so that we can easily hide within the cover media.

As shown in Figure 2, we could use local maxima and minima sampling data points to characterize the whole speech waveform. During our experimentation, we find that using local maxima and minima data points can achieve $\approx 80 - 85\%$ reduction in the number of samples required to reconstruct the speech waveform. It should be noted that the reconstructed waveform from the limited number of sample data points will be an approximate representation of the original waveform. In the encoding process, we will show how to embed local maxima and minima data points within the cover media samples.

3.3 Time Domain

The second challenge is to attach time values with extrema data points, i.e., to relate maxima and minima data points to their time of occurrence. The knowledge of this temporal relationship is necessary if we would like to reconstruct the speech waveform at the receiver end. To address this problem, we work on the frame level representing 160 sample data points. Every 20 ms, the sender sends an RTP packet to the receiver with the payload of 160 samples. While these frames (hence RTP packets) are created for the cover media, we analyze the covert media frame for its maxima and minima values. The cover and covert frames are of same size (i.e., with the same number of samples). We detect the extrema occurrences and their corresponding indices within the covert frame. Then, at the same index position within the cover frame, we replace its sample with the covert extrema sample.

3.4 Real-Time Presentation

For an effective two-way communication, it is necessary that the sender's audio should be rendered at the receiver end within 150 ms. Since our encoding and decoding processes are at the individual packet level, we are able to present media to the receiver player device well below the threshold of 150 ms.

4. ENCODING PROCESS

In this section, we describe the role of telephony codebook and how the covert media is hidden within the cover media and transmitted to the other end (i.e., callee-side).

In cryptography, a codebook is a document used for implementing a code. A codebook contains a lookup table for coding and decoding; each word or phrase has one or more strings which replace it. The codebook shown in Figure 3 is a set of rules represented as code index. The two communicating parties select and agree upon a particular rule to create (encode) and decipher (decode) the covert communication channel. Although the code index selection process between peers is beyond the scope of this work, it could be selected either during the signaling (i.e., call setup) phase or out of band.

As shown in Figure 4, there are two media sources. The cover media is sourced from a pre-recorded .wav file and the covert media, i.e., the user's utterances, is originated from microphone device. Both audio sources are sampled at the rate of $8K$ samples per second. The samples are stored in their respective input buffers. When a fixed number of samples (e.g., 160 samples) are collected, they will be available to the encoding module. Before sending to the encoding module, we analyze the fixed-duration covert frame

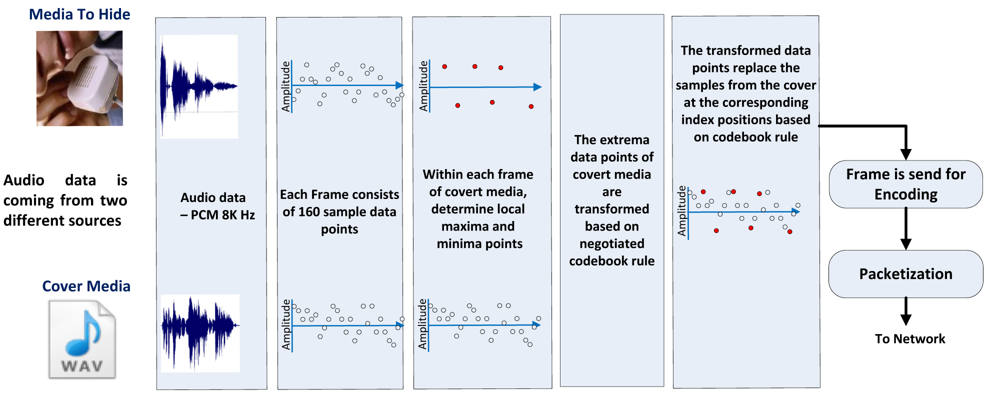
Media To Hide
Audio data is coming from two different sources
Cover Media
WAV
Audio data – PCM 8K Hz
Each Frame consists of 160 sample data points
Amplitude
Within each frame of covert media, determine local maxima and minima points
The extrema data points of covert media are transformed based on negotiated codebook rule
The transformed data points replace the samples from the cover at the corresponding index positions based on codebook rule
Frame is send for Encoding
Packetization
To Network

Figure 4: Encoding - Cover and Covert Media Samples are Interleaved Together

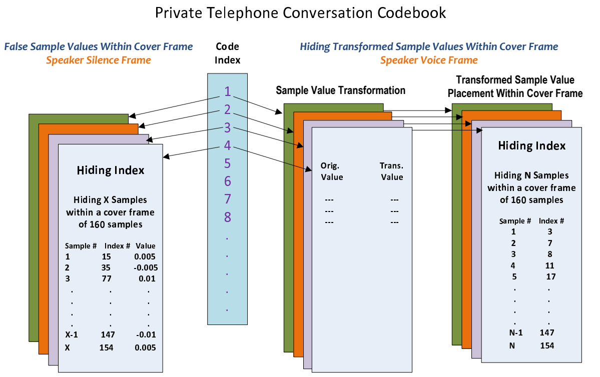
Private Telephone Conversation Codebook
False Sample Values Within Cover Frame
Speaker Silence Frame
Code Index
1
2
3
4
5
6
7
8
Hiding Transformed Sample Values Within Cover Frame
Speaker Voice Frame
Sample Value Transformation
Transformed Sample Value Placement Within Cover Frame
Hiding Index
Hiding X Samples within a cover frame of 160 samples
Sample # Index # Value
1 15 0.005
2 35 -0.005
3 77 0.01
X-1 147 -0.01
X 154 0.005
Orig. Value
Trans. Value
Hiding Index
Hiding N Samples within a cover frame of 160 samples
Sample # Index #
1 3
2 7
3 8
4 11
5 17
N-1 147
N 154

Figure 3: Telephone Codebook - Defines Rules as How the Samples of Covert Media are Transformed and also How to Interleave together with Cover Samples

to find local maxima and minima values and their corresponding index positions within 160 samples. Then, these extrema samples are transformed and replace the cover samples residing at the index positions as per selected codebook rule. It should be noted that the same cover media should not be reused for future communication. In the following, we briefly describe the encoding process in a step-by-step manner.

The codebook shown in Figure 3 is a document consists of a set of rules represented as code index. The codebook contains a lookup table for encoding and decoding. The two communicating parties select and agree upon a particular rule to create (encode) and decipher (decode) the covert communication channel.

4.1 Determine Local Maxima and Minima

The user's speech utterances over microphone generate voiced and unvoiced speech in succession, separated by silence regions. Though in the silence region, there is no excitation supplied to the vocal tract (i.e., no speech output), still silence is an integral part of speech signal. To hide as much information as possible about the user's speech within the cover media, we treat each frame differently depending upon its relevance to the voice activity or how important it is in the reconstruction of the waveform at the receiver end. Now assume that for a particular covert frame of n $(= 160)$ samples, the values are $y_1, y_2, ..., y_n$. First we determine the maximum (y_{max}) and minimum (y_{min}) values of the samples within the frame. If $y_{max} - y_{min} \leq 0.01$, it means that all the n samples of the frame are confined within a narrow band of 0.01 and we treat such a frame as silence without much voice activity. For silence covert frames, we do not determine local maxima and minima points; instead we choose X (very few, say 4-6) index positions within the cover frame and assign each one of them with a value as described in codebook rules (i.e., left side of Figure 3). Therefore, covert silence frames have insignificant impact on cover frames. For all the other covert frames with significant voice activity, we determine both local maxima and minima sample points.

Now assume that the value y_i is a local minimum if there exists a neighborhood y_{i-1}, y_i, and y_{i+1} with y_i = min$\{y_{i-1}$, y_i, $y_{i+1}\}$. However, there are a large number of local minima points and therefore to filter out insignificant local minima points, we extend the neighborhood to y_j, y_{j+1},...,y_i,y_{i+1},..., y_h with $j < i < h$ and consider only those y_i local minima points where $y_i = min\{y_j,...,y_h\}$, i.e., $y_i < y_k$ for $k = j, ..., i-1$ and $y_i < y_l$ for $l = (i+1), ..., h$. Similarly, we find y_i as a local maximum point in a list of sample values $y_1, y_2, ..., y_n$ if there exists a neighborhood y_j, y_{j+1},..., y_i, y_{i+1},..., y_h with $j < i < h$ such that $y_i = max\{y_j, ..., y_h\}$, i.e., $y_i > y_k$ for $k = j, ..., i-1$ and $y_i > y_l$ for $l = (i+1), ..., h$. In our experimentation, we find that the extrema value determination within the neighborhood of 3, 5, and 7 samples can achieve approximately 40%, 66%, and 82% reduction in the number of samples required to reconstruct the waveform, respectively.

4.2 Transform the Extrema Values

The local extrema values found in a covert frame are passed through a transformation process to modify its true value and scale down its range from $[+1.0, -1.0]$ to a much lower range of $[+0.1, -0.1]$ or $[+0.01, -0.01]$, with a reduction factor of 10 or 100, respectively. Now assume that $\mathbb{F}$ is a transformation function with Ψ as a desired reduction factor. Each code index has its own transformation function and reduction factor defined in the codebook rule (i.e., right side of Figure 3).

$$y_i \rightarrow [\mathbb{F}]_\Psi \rightarrow y'_i \quad \forall i, \text{ i.e., extrema indices}$$

Both the transformation functions and reduction factors are known to the receiver-side. Thus, based on the received transformed value y'_i, its corresponding true value y_i can be recovered.

For example, in our experimentation this transformation operation is implemented as a three-step process: first, we determine the sign (positive or negative) of the extrema sample value; secondly, the absolute value of the sample is passed through the transformation function $\ln|y_i|$; and thirdly, the natural logarithm of the

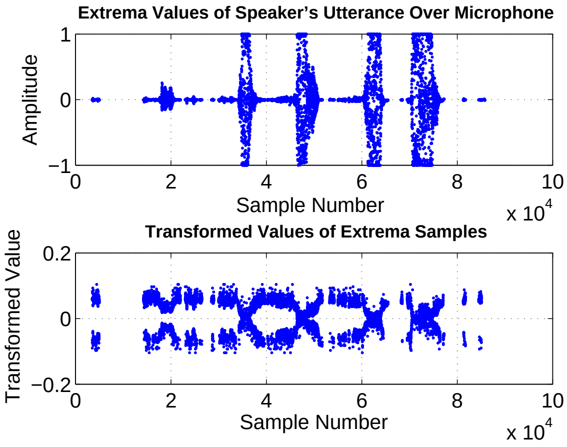

Figure 5: Transformation Operation on Extrema Sample

sample value is further reduced by a factor of $\Psi = 1000$ and then is assigned the same polarity as in the step one (i.e., y_i' = $\pm$ ln $|y_i|$ $/1000$). Figure 5 shows the transformation process where we apply the logarithm function and reduction factor on the original extrema samples, and thus transforming the true covert sample values before being transmitted within RTP payload.

The transformation process is applied on the extrema values mainly because of the following two reasons: (1) lower covert sample values will appear as a soft noise that can easily be hidden within the cover media's background noise when the cover media is intercepted and decoded by eavesdroppers; (2) even if a sample value is guessed, still it will be difficult to uncover the true value of the covert sample, making it very hard to reconstruct the covert samples and listen into the real content of a voice conversation. Therefore, the real communication remains private between two peers.

4.3 Determine Hiding Location

Now we need to determine some hiding locations and let the receiver-end be aware of them, so that the receiver can know exactly where to search for the hidden covert samples within a frame. There are many possibilities of selecting a hiding pattern. For example, as shown in Figure 3, these hiding locations can be selected as any arbitrary indexes within the frame. The codebook-based hiding locations achieve three features: first, it makes the implementation of decoding process very simple; secondly, it makes eavesdroppers exceedingly difficult to perform statistical analysis and find hiding patterns; and finally, by having the knowledge of hiding locations, we can still recover the covert sample values even if the RTP stream may have undergone through the transcoding process.

4.4 Swap Cover Samples

In this step we interleave both covert and cover samples together. The transformed extrema values of the covert media frame replace the samples of the cover media frame at the same corresponding indices based on the codebook rule negotiated between peers. If we assume that y_i' is an i^{th} (e.g., say 5^{th}) transformed covert sample, then after consulting the codebook rule as shown in Figure 3, we find that it will swap with the j^{th} (e.g., 17^{th}) sample Y_j of the cover frame, i.e., $Y_j \leftarrow y_i'$, ∀i. Once the samples from the covert and cover media are interleaved together within a frame, they will be sent to the encoding process based on the codec negotiated during the initial call setup phase. The compressed audio data is packetized as RTP media packets and transmitted to the destination.

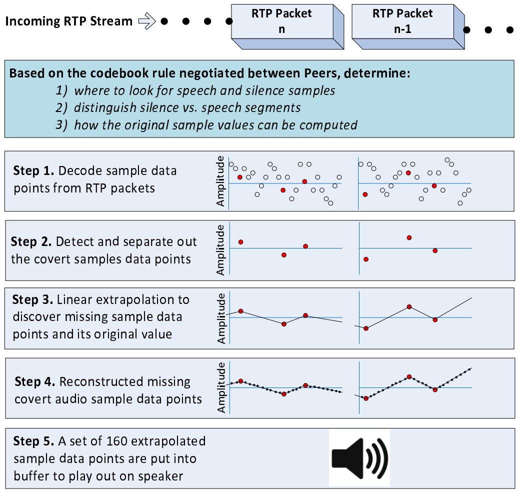
Incoming RTP Stream
RTP Packet
n
RTP Packet
n-1
Based on the codebook rule negotiated between Peers, determine:
1) where to look for speech and silence samples
2) distinguish silence vs. speech segments
3) how the original sample values can be computed
Step 1. Decode sample data points from RTP packets
Amplitude
Step 2. Detect and separate out the covert samples data points
Amplitude
Step 3. Linear extrapolation to discover missing sample data points and its original value
Amplitude
Step 4. Reconstructed missing covert audio sample data points
Amplitude
Step 5. A set of 160 extrapolated sample data points are put into buffer to play out on speaker

Figure 6: Decoding - Hidden Samples are Extracted and Extrapolated to Find Missing Samples

5. DECODING PROCESS

Once the G.711 encoded audio RTP stream arrives at the receiver side, the packets are validated for their correctness and the arrival time is also noted (for jitter, clock skew estimation etc.). Based on RTP time-stamps, these packets are ordered and added to an input queue. Then, the packets are extracted from the input queue and inserted into a source-specific playout buffer. The frames are held in the playout buffer for a period of time to smooth timing variations caused by the networks. In the very last processing stage, the frames are decompressed and rendered for users. Our decoding process is implemented at this stage, a set of processing steps are executed on the decompressed data to extract the hidden audio samples.

The decoding process is based on prior knowledge of what codebook rule is negotiated between peers. The codebook rule informs us as how to detect and distinguish covert samples from cover samples and also how to recover the original sample values from the received modified sample values. As shown in Figure 6, the decoding process can be described as a five step process. In the very first step, we get decompressed samples within a particular frame, say n^{th} time frame. In the second step, we detect and extract the hidden samples at particular index values (based on the codebook rule). The list of hiding indexes (i.e., the sample positions within a frame) for speech and silence is known between the sender and receiver. The received frame is checked for some particular index positions and their corresponding values to discover a pattern matching with the silence frame. If a pattern is found then the entire received frame is discarded and a new reconstructed frame of 160 samples[2] (all with 0.00 value) is put into play out buffer. Now assume that the n^{th} time frame is a speech frame (i.e., a frame carrying covert speech samples). Following the codebook rule, we discover k data points (i, y_i'), where i is the index value and y_i' its corresponding value at i^{th} index. We maintain a list of $key - value$ pairs for all the discovered data points in the format $[i, y_i']$. It should be noted that the i^{th} sample value may not be the true value of the hidden sample. The sender-side may have modified the true value by pass-

[2] Here we assume that each G.711 packet is worth of 20 ms audio with 160 samples. We can accommodate any changes in the packet size (such as 10 or 30 ms) if knowing the codec and its attributes negotiated between peers.

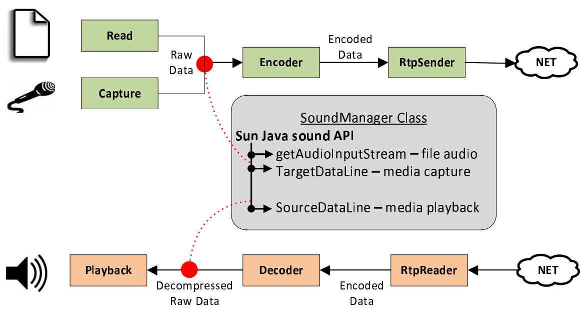

Figure 7: Encoding and Decoding Process Within a Java-based Peers SIP Client

ing it through an operation known to the sender and receiver only. Therefore, at the receiver side knowing the modifying parameter (i.e., Ψ), and the transformation function (i.e., $\mathbb{F}$), we can extract the sample's original value $y_i \leftarrow y_i'$.

In the third step, we apply linear extrapolation to recover missing samples. Using each pair of conjugate data points (such as i' and i), we calculate the slope ($m_i = \frac{y_i - y_{i'}}{i - i'}$) and intercept ($c_i = y_i - m_i * i$). In the fourth step, using the slope and intercept values, we estimate all the other missing samples between index i' and i. In this way, we can estimate $(i - i')$ samples for a domain of index values $[i', i)$. The third and fourth steps repeat for each pair of the conjugate data points in the list. In the fifth step, once 160 samples are estimated, we put them in device input buffer for playing.

6. EXPERIMENTAL EVALUATION

We implemented a prototype of the proposed wiretap-proof approach and deployed it in software-based phone clients. Then, we conducted a series of realistic experiments to validate its effectiveness. In our experimental testbed, two computers are used as the SIP-based telephones, and communicate with each other over IP networks. Their IP telephony service is provided by the same popular VoIP service provider. The service provider's SIP server and SBC are located in Greenville, SC. Both of the telephone endpoints, i.e., the computers, have 2.26 GHz Intel Core2Duo and 4 Gbytes of RAM, running Windows Vista OS and are connected to the Internet via cable modems. Both of the SIP clients are located in Aldie, VA, and the calls are routed through the Internet, while the service provider's network edge device (i.e., SBC) is located at 13 hops away with the average round trip time of 38 ms.

6.1 Softphone Implementation

Using the computer's audio system and microphone, a PC-based softphone, i.e., a software-based phone client, works as a regular telephone to place and answer phone calls. Our encoding and decoding processes are implemented with the publicly available open source Java-based SIP client known as *Peers* [1]. Throughout our implementation, we assume the following audio format:

```
// linear PCM 8kHz, 16 bits, mono, signed, little endian
audioFormat = new AudioFormat(8000, 16, 1, true, false);
```

For two-way communication, both encoding and decoding modules are implemented on a SIP client.

For the media handling, `IncomingRtpReader` and `CaptureRtpSender` are included in Peers as the two main classes. The `IncomingRtpReader` is responsible for RTP depacketization, media decompression, and media playback; and `CaptureRtpReader` is responsible for microphone capture, media encoding, and RTP packetization. However, for media processing, the whole media package relies on standard *Sun Java Sound API*. The Peer's `SoundManager` class implements its all interaction with the Java Sound API. Within this class, we have references for both *SourceDataLines* for media playback and *TargetDataLine* for media capture. Since our encoding and decoding processes are based on the raw data as shown in Figure 7, most of our software implementation is done within this class.

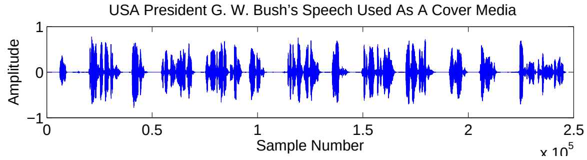

Figure 8: Cover Media

6.2 Experimental Results

In our experiments, for cover media we used both music clips and pre-recorded human conversation files, though in real life, it is preferable that caller should use his own earlier prerecorded telephone conversations. As an example, in Figure 8 we show a 30 sec. snippet of cover media source that is a pre-recorded speech of the former President G.W. Bush given in the eve of September 11^{th} terrorist attack [22]. We selected this particular media file because it represents both silence and speech segments with the true nature of two-way communication (i.e., speech on and off periods).

The speaker's utterances over microphone are captured and sampled, and then the fixed size frames are created. Figure 9 (top) shows the extrema samples of the speaker's utterances *"Hello! How are you?"* over microphone within about 30 seconds call duration. It should be noted that the encoding process is per frame based, though the Figure shows the whole 30 seconds call duration. Figure 9 (bottom) shows how the extrema samples are transformed to some other values using a transformation function known to the receiver-side as well. For easy presentation, here we used $\ln |y_i|$ $/1000$ transformation operation to modify the true values of the extrema samples (i.e., y_i), although we could transform these samples in many different ways. The transformed values replace the cover media samples at the corresponding indices as indicated in the negotiated codebook rule during the call setup phase. In this way, we interleaved both covert and cover media samples together. The interleaved samples are packetized and transmitted over the Internet to the VoIP service provider, and then get relayed to the receiver-side phone.

The interleaved media packets can be captured by eavesdroppers between the caller and callee phones. We intercepted the me-

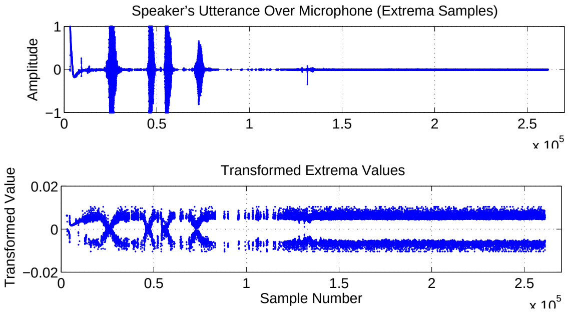

Figure 9: Original Extrema Samples (top) and its Transformed Values (bottom)

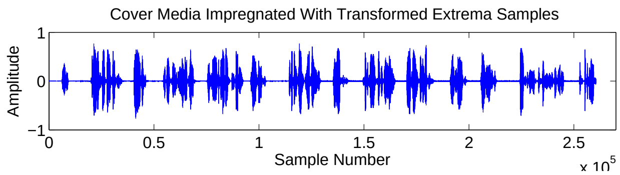

Figure 10: Cover Carrying Covert Media

dia packets at the SBC located at the edge of the VoIP service provider's network. The media packets are decoded and played using Wireshark tool. Figure 10 shows the intercepted media that is impregnated with covert samples. When we decode the audio, we are able to hear the cover media only.

Now our task is to measure the perceived audio quality. The Mean Opinion Score (MOS) is the most widely accepted measure of the perceived quality. However, it is generated by averaging the results of a set of standard, subjective tests where a number of listeners rate the heard audio quality. In our case, we are more interested in a quantitative measure based on algorithm. We used Perceptual Evaluation of speech (PESQ) [27], an International Telecommunication Union (ITU) standard. The algorithm estimates the perceived quality difference between the test and the original audio signal. The PESQ score is in the range of -0.5 to 4.5. For most cases, the audio signal with listening quality has a score between 1 to 4.5. In our experiments, we assumed the original cover media as a reference with the score of 4.5, and then we measured the perceived audio quality of interleaved audio signals. Based on experimental results, we found that the interleaved test audio signals have PESQ score of 1.7 to 2.3. The voice quality is high enough for human users to easily listen[3].

The analysis of received RTP stream is tabulated in Table 1 as experiment number 1. The media session duration is about 30 seconds, in which we captured 1635 RTP packets (one-way direction). We observed that the encoding process induces insignificant delay to individual packet departure time and the end point's perception of this call remains the same as any other normal calls.

Table 1: **Analysis of RTP Stream (One Way)**

Exp. Number	Total Packets	Max Jitter	Mean Jitter	Lost Packets	Sequence Error	PESQ Score
1	1635	3.10 ms	1.18 ms	0	0	1.75
2	2301	6.16 ms	1.08 ms	0	0	1.94
3	3346	1.97 ms	1.08 ms	0	0	2.18
4	6080	1.72 ms	0.95 ms	0	0	1.72
5	9107	2.68 ms	1.03 ms	0	0	1.89

At the receiver side, the decoding module of the softclient receives the decompressed samples from the interleaved media stream and tries to separate the covert samples from the cover media based on the already known hiding pattern. In our software implementation, during the speaker's silence period we are not able to detect the silence pattern precisely enough, resulting in false data points and thus introducing noise in the reconstructed audio signals. Both false and true discovered samples are reversely transformed to their original values, and all the missing samples are estimated using linear interpolation method. Figure 11 shows the reconstructed waveform based on the limited number of covert samples mixed with false samples. We performed quantitative measure of the perceived quality of the reconstructed audio signals. When we compared the speaker's utterances over the microphone to the receiver's perception as what is played out of the speaker, the PESQ score is 1.1 to 1.3. Thus, it still remains an audible quality even in the presence of false data points as noise.

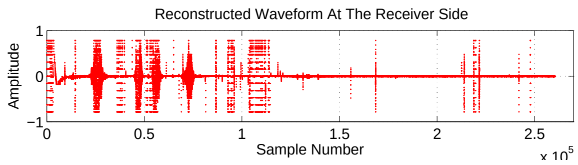

Figure 11: Reconstructed Audio Waveform from Discovered Covert Samples

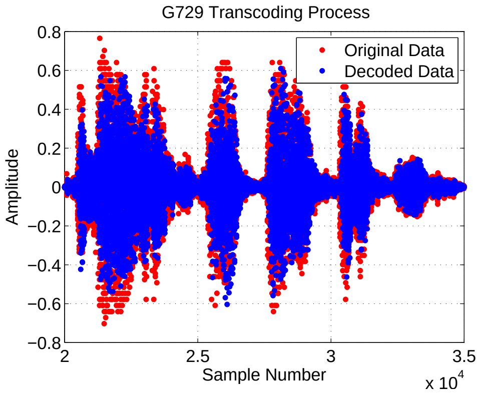

Figure 12: Sample Values Modified by Transcoding Process

6.3 Transcoding Scenario

During the initial growth and deployment, most of the VoIP service providers standardized on `G.711` codec. It became a de facto codec choice because of its widespread use in telecommunication, low processing overhead, associated licensing cost, and high voice quality. However, `G.711` requires a relatively high amount of bandwidth. In many cases where customers have limited upstream bandwidth (e.g., DSL deployments), they are forced to choose codecs requiring lower bandwidth such as `G.729`. Within VoIP networks, it is possible that individual end points (i.e., phones) may be using different codecs. The codec transcoding provides a way to change one codec format to others and vice versa, without making codec change on individual end points. Generally, transcoding is done at the edge of a service provider network on media gateways or session border controllers. Being proprietary in nature and involved licensing cost, we did not implement `G.729` codec within Peers SIP clients. In our experiments, the `G.711` audio stream was captured using wireshark tool and then the RTP payload raw data was transcoded to `G.729` codec format using *VoiceAge Open `G.729` Implementation* [4]. In order to recover the covert channel, we again transcode (or decode) the `G.729` encoded raw data back to PCM format `G.711` data. Figure 12 shows both the original `G.711` RTP stream data points and the transformation to sample data point values after applying `G.729` encoding and decoding processes.

Here we can clearly see why the conventional audio steganography approaches do not survive the transcoding process. This is because almost all of the sample data points are modified to some new values. However, since our proposed approach is based on interleaving of the cover and covert media samples, it can preserve their relative magnitudes; therefore, it is still possible to reconstruct

[3] Many compressed codecs have PESQ scores around 3.7.

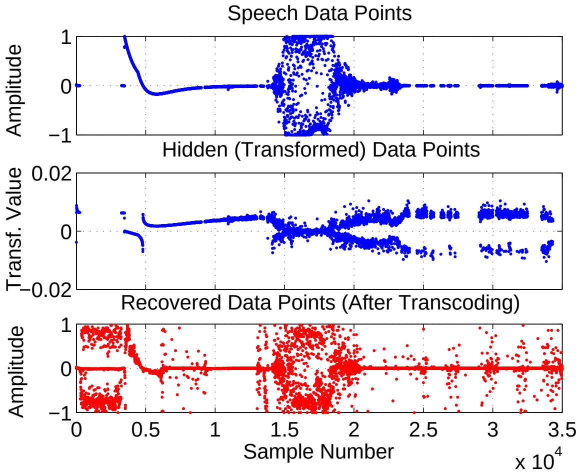

Figure 13: Recovering Covert Data Points After Decoding `G.729` Audio

the covert channel. The extrema data points of speaker's utterance *"Hello!"* over microphone is shown at the top part of Figure 13. The middle portion of Figure 13 shows the transformed values of the extrema samples that are interleaved with the cover samples. Later interleaved samples are encoded into `G.729` format. For covert channel recovery, we transcode it again back to `G.711` PCM data. Once the decoded data points are obtained from the `G.729` encoded audio, we apply the same procedure as described in Section 5. The lower part of Figure 13 shows the recovered covert samples. Because of the modified sample values, many of the hidden patterns meant to indicate silence segments were lost. These wrongly identified segments are treated as speech and when extrapolated to create missing samples cause noise.

6.4 Further Discussion

Since the proposed method does not use least significant bits of the samples or any previously known audio steganography methods, qualitatively our approach remains undetectable to current genre of LSB-based steganalysis tools. Now we discuss more intuitive approaches to identify and discover the presence of covert channel.

6.4.1 Visual Effect:

Since our approach is based on modifying whole samples, rather than bits, it seems that the proposed approach could be detected by examining the waveforms. We performed a set of experiments comparing both LSB-based methods and our proposed codebook-based sample replacement method. In the same cover media, we hide a line of text using Steghide [3] - a LSB-based tool and embed speaker's utterance *"Hello!"* using codebook-based sample replacement method, respectively. However, if we look at Figure 14, we cannot see any significant difference in the waveforms, and cannot guess there are different messages hidden in them.

6.4.2 Statistical Analysis:

To perform statistical analysis, we created 200 bins of equal size of 0.01 covering the sample value range of [-1.0, 1.0]. Figure 15 shows histograms and the distribution of bins for three separate cover medias. Based on the analysis results in Table 2, we can see that LSB-based media is almost identically distributed as the original cover media. The codebook-based method has higher mean compared to LSB-based media. However, it should be noted that without the knowledge of original cover media, it is hard to differentiate and detect the presence of covert channel.

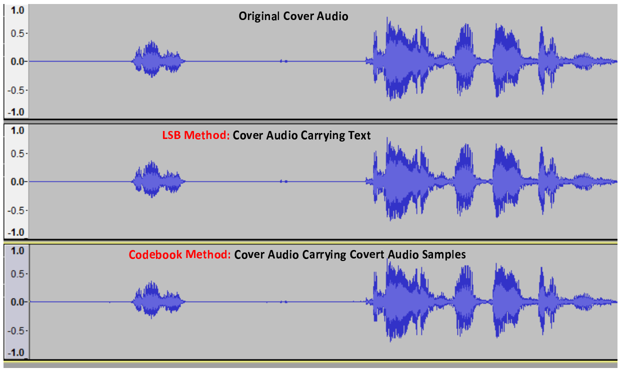
1.0
0.5
0.0
-0.5
-1.0
Original Cover Audio
LSB Method: Cover Audio Carrying Text
Codebook Method: Cover Audio Carrying Covert Audio Samples

Figure 14: Visual Waveform Comparison

Table 2: Statistical Analysis of Samples

	Cover	LSB	Codebook
No of Samples	80,000	80,000	80,000
Mean	1.540e-05	1.538e-05	3.513e-04
Std Dev.	0.1090	0.1090	0.0999

6.4.3 Voice Quality:

The voice quality of cover media is almost unaffected by LSB-based audio steganography. Within the 10 second cover media we hid two lines of text message, the PESQ score of LSB-based was still 4.48. By contrast, the proposed codebook-based sample replacement method has the PESQ score of about 2.0. When we hear intercepted media, there is always a soft-noise in the background. The untrained or casual ears cannot make any distinction, however algorithm-based quality monitoring tools can detect such noise easily. However, the fundamental question is *does the low voice quality mean presence of covert channel?* The short answer is "No", as voice quality also heavily depends upon the speaker's ambient environment. Moreover, even if we could guess the presence of covert channel, it is still very hard to find covert samples (i.e., hiding indexes within a frame) and its original value (i.e., transformation functions and its parameters) to reconstruct the waveform.

7. RELATED WORK

There are various malicious activities against telephony systems and their users. Most of these attacks are trying to exploit the insecure or poorly protected systems and to eavesdrop on unencrypted network traffic. Even the encrypted contents of VoIP traffic can be exposed by comparing packet size and interarrival times to language constructs [31, 32]. As VoIP has been increasingly adopted for communication in enterprises and organizations, the data exfiltration by covert channels piggybacking on VoIP signaling and media sessions has become a serious threat. Wojciech et al. [13] exploited free/unused protocol fields and used intentionally delayed audio packets to create a covert channel. Takhiro et al. [21] discussed various audio steganographic and watermarking techniques that could be used to create VoIP covert channels. Huang et al. [12] presented a novel high-capacity steganography algorithm for embedding data in the inactive frames of low bit rate audio streams encoded by `G.723.1` source codec.

The CALEA infrastructure is vulnerable to malicious attacks. Lawful interception requires the telephone service providers to al-

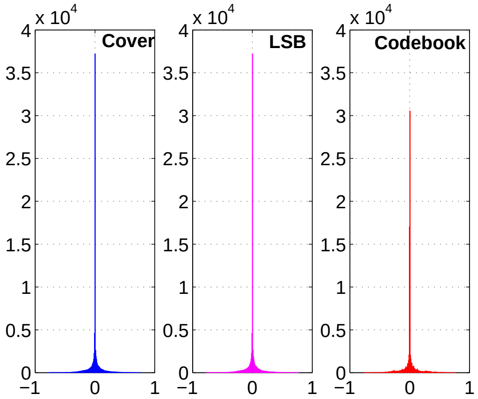

Figure 15: Statistical Analysis

low access to some particular target's call content and its related meta data. However, Sherr et al. [17] demonstrated the ability to prevent call audio from being recorded by injecting in-band signaling tones into a conversation. It is also possible that an attacker can overload the wiretapping system and prevent critical information to be logged [18]. Recently, Bates et al. [5] proposed an accountable wiretapping architecture that enhances wiretapping systems by adding tamper-evident records of wiretap events. However, our proposed work demonstrates that an attack can work in a very stealthy manner and then circumvent the existing accountable wiretapping infrastructure.

8. CONCLUSION

In this paper, we have presented a new audio steganography approach to create a real-time covert communication channel within another real-time media session. Our approach exploits the approximate audio signal construction to hide and recover voice information. Through real experiments, we have shown that even if we have only a few key characteristic samples of the waveform, a good approximate audio signal can still be recreated. By interleaving the cover and covert media samples, we can construct a real time covert voice channel and ensure two parties to have a regular phone conversion but in a secure and private fashion. Our study will expose a serious challenge to the media interception techniques used by law enforcement agencies. We hope our proposed wiretap-proof approach will motivate researchers and practitioners to further evaluate the security of the deployed interception systems and perform the vulnerability assessment of covert channels that could be created within media streams.

9. ACKNOWLEDGMENT

We would like to thank the VoDaSec team for their insightful feedback and comments. This work was partially supported by ARO grant W911NF-11-1-0149 and ONR grant N00014-13-1-0088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

10. REFERENCES

[1] Peers Softphone. Website, http://peers.sourceforge.net/, 2013.

[2] S-Tools 4.0. Website, http://www.spychecker.com/program/stools.html, 2013.

[3] Steghide. Website, http://steghide.sourceforge.net/, 2013.

[4] VoiceAge – Open G.729. Website, http://www.voiceage.com/openinit_g729.php, 2013.

[5] A. Bates, K. Butler, M. Sherr, C. Shields, and P. T. M. Blaze. Accountable Wiretapping -or- I know they can hear you now. In *NDSS*, 2012.

[6] D. McCullagh. Bin Laden: Steganography Master? Website, http://www.wired.com/politics/law/news/2001/02/41658?currentPage=all, 2012.

[7] O. Dabeer, K. Sullivan, U. Madhow, S. Chandrasekeran, and B. S. Manjunath. Detection of hiding in the least significant bit. In *In IEEE Trans. on Signal Processing*, pages 3046–3058, 2004.

[8] Fabien Petitcolas. mp3stego. Website, http://www.petitcolas.net/fabien/steganography/mp3stego/, 2013.

[9] J. Fridrich, M. Goljan, and R. Du. Reliable Detection of LSB Steganography in Color and Grayscale Images. *IEEE Multimedia*, 8:22–28, 2001.

[10] Gary C. Kessler. Steganography: Hiding Data Within Data. Website, http://www.garykessler.net/library/steganography.html, 2012.

[11] Heinz Repp. Hide4PGP. Website, http://www.heinz-repp.onlinehome.de/Hide4PGP.htm.

[12] Y. F. Huang, S. Tang, and J. Yuan. Steganography in Inactive Frames of VoIP Streams Encoded by Source Codec. *IEEE Transactions on Information Forensics and Security*, 6:296–306, 2011.

[13] W. Mazurczyk and K. Szczypiorski. Steganography of VoIP streams. *CoRR*, abs/0805.2938, 2008.

[14] N. Provos and P. Honeyman. Hide and seek: An introduction to steganography. *IEEE Security and Privacy*, 1(3):32–44, May-June 2003.

[15] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, IETF Network Working Group, 2002.

[16] l)ruid. DEFCON 15: Real-time Steganography with RTP. Website, http://www.youtube.com/watch?v=boTZ0ZAcF5I.

[17] M. Sherr, E. Cronin, S. Clark, and M. Blaze. Signaling Vulnerabilities in Wiretapping Systems. *IEEE Security & Privacy*, 3(6):13–25, 2005.

[18] M. Sherr, G. Shah, E. Cronin, S. Clark, and M. Blaze. Can they hear me now?: a security analysis of law enforcement wiretaps. In *ACM Conference on Computer and Communications Security*, 2009.

[19] K. Sullivan, O. Dabeer, U. Madhow, B. Manjunath, S. Chandrasekaran, and S. Chandrasekeran. LLRT Based Detection of LSB Hiding. In *In Proceedings of ICIP*, pages 497–500, 2003.

[20] T. Mogg . Anonymous hacks call between FBI and Scotland Yard about hackers. Website, http://www.digitaltrends.com/international/anonymous-hacks-call-between-fbi-and-scotland-yard-about-hackers, 2012.

[21] T. Takahashi and W. Lee. An assessment of VoIP covert channel threats. In *SecureComm*, 2007.

[22] The Bots. The George W. Bush Public Domain Audio Archive. Website, http://www.thebots.net/GWBushSampleArchive.htm, 2012.

[23] C. Wang and Q. Wu. Information Hiding in Real-Time VoIP Streams. *Proceedings of the Ninth IEEE International Symposium on Multimedia*, pages 255–262, 2007.

[24] A. Westfeld and A. Pfitzmann. Attacks on Steganographic Systems. In *Information Hiding*, pages 61–76, 1999.

[25] Wikipedia. CALEA. Website, http://en.wikipedia.org/wiki/Communications_Assistance_for_Law_Enforcement_Act, 2013.

[26] Wikipedia. Greek wiretapping case. Website, http://en.wikipedia.org/wiki/Greek_wiretapping_case_2004%E2%80%932005, 2013.

[27] Wikipedia. PESQ. Website, http://en.wikipedia.org/wiki/PESQ, 2013.

[28] Wikipedia. Raj Rajaratnam. Website, http://en.wikipedia.org/wiki/Raj_Rajaratnam, 2013.

[29] Wikipedia. Rod Blagojevich. Website, http://en.wikipedia.org/wiki/Rod_Blagojevich, 2013.

[30] Wikipedia. Steganography. Website, http://en.wikipedia.org/wiki/Steganography, 2013.

[31] 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 *IEEE Symposium on Security and Privacy*, 2008.

[32] C. V. Wright, L. Ballard, S. E. Coull, F. Monrose, and G. M. Masson. Uncovering Spoken Phrases in Encrypted Voice over IP Conversations. *ACM Trans. Inf. Syst. Secur.*, 13(4):35, 2010.