SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1500

SPECIAL ISSUE PAPER

Micro protocol engineering for unstructured carriers: on the embedding of steganographic control protocols into audio transmissions

Matthias Naumann[1], Steffen Wendzel[2]*, Wojciech Mazurczyk[3] and Joerg Keller[1]

[1] University of Hagen, Germany
[2] Fraunhofer FKIE, Germany
[3] Warsaw University of Technology, Poland

ABSTRACT

Network steganography conceals the transfer of sensitive information within unobtrusive data in computer networks. So-called micro protocols are communication protocols placed within the payload of a network steganographic transfer. They enrich this transfer with features such as reliability, dynamic overlay routing, or performance optimization — just to mention a few. We present different design approaches for the embedding of hidden channels with micro protocols in digitized audio signals under consideration of different requirements. On the basis of experimental results, our design approaches are compared and introduced into a protocol engineering approach for micro protocols. Copyright © 2016 John Wiley & Sons, Ltd.

KEYWORDS
network steganography; audio steganography; micro protocols; control protocols; covert channels

***Correspondence**
Steffen Wendzel, Department of Cyber Security, Fraunhofer FKIE, Friedrich-Ebert-Allee 144, D-53113 Bonn, Germany.
E-mail: steffen.wendzel@fkie.fraunhofer.de

1. INTRODUCTION

Network steganography is the art and science of transferring secret information over a network while the hidden transfer itself is concealed. Network steganography is applied when the encryption of secret data is not sufficient [1]. For instance, journalists can use network steganography to keep the transfer of illicit information hidden when facing Internet censorship. In addition to the transfer of hidden payload, so-called micro protocols (or control protocols) can be embedded into a network steganographic communication. Micro protocols enrich the feature-set of the steganographic channel in various aspects [2], such as

- adapting the steganographic channel's configuration automatically to bypass new Internet censorship infrastructure,
- integrate functionality for reliability and segmentation of hidden data, or
- provide dynamic overlay routing and optimization for the stealthiness of a hidden transmission.

Being steganographic payload itself, a poor micro protocol design and placement can lead to a higher detectability of a steganographic communication. This problem led to the development of protocol engineering methods for micro protocols by Wendzel, Keller, and Backs in [2–4]. These protocol engineering methods either minimize the size of the micro protocol or the placement within a *structured* carrier (machine readable, such as protocol header fields) but not the placement within an *unstructured* carrier (understandable by humans, such as audio content in audio or video transmissions).

Our work presents the first design approaches, evaluation, and protocol engineering approach for micro protocols to be placed in unstructured carriers. We demonstrate our approach on the basis of digital audio transmissions.

Micro protocols comprise their own protocol headers. These headers can be designed in two ways, either in form of a

(1) *static header*, that is, the whole header is present in one chunk, or in form of a
(2) *dynamic header*, that is, the header can be split into separate parts, which are included on demand and spread over the carrier.

For our micro protocol design, we consider both static and dynamic headers. Moreover, our work examines how micro protocols must be designed to provide a high level of covertness (stealthiness) in unstructured carriers. For the application of micro protocols to hidden data within audio streams, criteria like covertness, capacity, and robustness of the transmitted data are considered with respect to the specific conditions of a transmission of audio data and the human auditive perception.

The reminder of this paper is structured as follows: Section 2 explains fundamentals of related topics and discusses related work on micro protocols. In Section 3, we perform the requirements analysis for our micro protocol engineering approach, followed by Section 4 in which we discuss the implementation of the approach. Section 5 evaluates our work on the basis of experiments. On the basis of this evaluation, Section 6 presents our protocol engineering approach. Section 7 concludes and discusses future work.

2. FUNDAMENTALS

In this section, we first introduce basics of network steganography and related topics, which are of importance for this paper. We finish with an introduction to micro protocols and a discussion of related work.

2.1. Steganography and its detection

The goals of a steganographic transfer can vary from the theft of confidential data, for example, within industrial espionage, over the hidden communication of command and control data in botnets, to the transmission of illicit information by journalists [1]. Steganography dates back to ancient Greece. Early approaches for steganography included the hiding of information using, for example, an invisible ink. In the late 20th century, digital media steganography arose, which focuses on hiding information in digital audio, image, and video files [5].

In *digital audio steganography*, confidential data is embedded into digital sound. Therefore, bit sequences of the digital audio files are slightly changed to encode hidden data. This embedding of data into audio files is especially challenging because the human auditive perception is highly sensitive, in particular, even more sensitive than the human visual system [6]. Research focuses on the development of suitable technologies to prevent the perception of the steganographic data in audio files by both, humans and machines. In [7], some of the most widely used algorithms for audio steganography are discussed, which are least significant bit (*LSB*) Coding, Phase Coding, Parity Coding, Spread Spectrum, and Echo Data Hiding. Transferring hidden data in audio data shares some aspects with audio watermarking (cf. e.g., [8]), such as close to no influence on perception by humans and survival of operations like filtering. The difference is that audio watermarks are not necessarily hidden data and that they are contempt if enough data from the watermark remains to identify the watermark, while enough of the hidden data (including error-correcting code) must remain in order to completely and correctly decode the payload.

Network steganography stealthily transfers arbitrary data over a network. Therefore, either structured or unstructured carriers can be used. *Structured* data are those interpretable by machines, having a pre-defined structure, such as a network protocol header, while *unstructured* data are interpretable by a human, for example, the content of an audio stream, an image file, or a natural language [9]. For instance, if data is hidden within the IPv4 TTL field, it is hidden in structured data because protocol headers always have a pre-defined, machine-interpretable structure. The embedding of hidden data in inaudible parts of audio streams, on the other hand, is a form of network steganography in unstructured data.

From the audio steganography perspective, many methods (both for data hiding as well as for their detection) have been designed and investigated, for example, for services like Voice over IP (VoIP). As in this paper, we investigate micro protocols for VoIP-like service that utilize the LSB algorithm as an underlying steganographic method; thus later, we review related work that describes LSB-based solutions and their detection. Currently, a lot of research effort is still devoted for improving LSB steganography for real-time services like VoIP. The first VoIP steganographic method to utilize the digital voice signal as a hidden data carrier was proposed by Aoki in [10]. The LSB steganography was utilized to provide a packet loss concealment method for G.711-based VoIP. Dittmann *et al.* in 2005 presented the first VoIP steganography implementation prototype that also used the LSB method [11]. Wu and Yang described the scheme of adaptive LSB [12], which for G.711-based speech calculates energy statistics to estimate the number of LSBs to be utilized as a hidden data carrier in each voice sample. The results proved that this approach performs better than simple LSB and offers a higher steganographic bandwidth (about 20 kbit/s) while introducing less degradation of the voice quality. Wang and Wu also suggested using the LSBs of voice samples to carry secret communication [13], but in their solution, the bits of the steganogram were coded using a low rate voice codec, like Speex. Their prototype implementation is characterized by a small processing delay of about 0.257 ms. Takahashi and Lee presented a proof of concept LSB-based implementation, Voice over VoIP (Vo2IP), which is able to establish a hidden communication by embedding 8 kbit/s G.729-based compressed voice data into the regular pulse code modulation-based voice traffic [14]. In 2008, Liu *et al.* found that the LSBs of each speech frame for G.729 can be replaced with secret data bits [15]. The experimental results indicate that the method is perceptually transparent while the steganographic bandwidth is relatively high (about 200 bit/s). Tian *et al.* proposed the use of an LSB steganography-based system that employs a well-balanced and simple encryption of secret data [16]. This system was evaluated for VoIP with G.729a speech coding using a proof of concept tool named *StegTalk*. The experimental results showed that the achievable steganographic bandwidth is in the range 0.8 – 2.6 kbit/s and has a negligible effect on speech quality. Moreover, it met the real-time requirements of the VoIP service. A real-time steganography system for VoIP was described by Tian *et al.* in [17]. The main novelty of the proposed solution is not in the steganographic method used (LSB), but in utilizing M-sequence encryption techniques to eliminate the correlation among secret messages to increase the resistance against statistical steganalysis. Moreover, protocol steganography (usage of free/unused fields in protocols headers) is applied to provide a novel synchronization mechanism together with an RSA-based key agreement that ensures accurate restitution of the secret messages on the receiver side. This system was experimentally evaluated for 0.8 and 2.6 kbit/s steganographic bandwidth, which obtained a 0.3 and 1 quality drop in the mean opinion score (MOS) scale (which is typically used for expressing the quality of VoIP calls), respectively, and the total embedding latency increased by about 4.7 ms when 1 MB of steganogram is transmitted. In 2009, Xu and Yang proposed an LSB-based method dedicated to voice transmission using the G.723.1 codec in 5.3 kbit/s mode [18]. They identified five LSBs of the line spectrum pair vector quantization indices and used them to transmit hidden data; the method provided a steganographic bandwidth of 133.3 bit/s. An insightful overview of the general techniques that can be applied to VoIP steganography methods to make their detection even more difficult was introduced by Tian *et al.* in [19]. Additionally, they proposed three new encoding strategies based on digital logic. All techniques were evaluated for LSB-based steganography and proved to be effective. Another adaptive LSB-based steganography approach named *Adaptive VoIP Steganography* was proposed by Xu *et al.* [20]. Adaptive VoIP Steganography has two components: *Value-based Multiple Insertion*, which is responsible for dynamically selecting multiple bits based on the VoIP vector value and *Voice Activity Detection Dynamic Insertion*, which dynamically changes the embedding intervals to make detection harder. The approach was implemented for G.711-based VoIP, and the results prove that it is less detectable than a classic LSB method while achieving a steganographic bandwidth of about 114 B/s, introducing acceptable delay and degrading the voice from 0.1 to 0.4 on MOS scale. Finally, Liu *et al.* adopted least-significant-digits rather than LSBs to hide secret data [21]. This approach can increase around 30% of steganographic bandwidth while introducing lower steganographic cost than classic LSB method.

From the covert communication detection perspective statistical steganalysis for LSB-based VoIP steganography was first proposed by Dittmann *et al.* [11]. They proved that it was possible to detect hidden communication with almost a 99% success rate under the assumption that there are no packet losses and the steganogram is unencrypted/uncompressed. Takahasi and Lee described a detection method based on calculating the distances between each audio signal and its de-noised residual by using different audio quality metrics [14]. Then, a support vector machines classifier is utilized for detection of the existence of hidden data. This scheme was tested on LSB, Direct Sequence Spread Spectrum, Frequency Hopping Spread Spectrum and Echo hiding methods, and the results obtained show that for the first three algorithms the detection rate was about 94% and for the last it was about 73%. A Mel-Cepstrum based detection, known from speaker and speech recognition, was introduced by Kraetzer and Dittmann [22] for the purpose of VoIP audio steganalysis. Under the assumption that a steganographic message is not permanently embedded from the start to the end of the conversation, the authors demonstrated that the detection of an LSB-based steganography is efficient with a success rate of 100%. Steganalysis of LSB steganography based on a sliding window mechanism and an improved variant of the previously known Regular Singular algorithm was proposed by Huang *et al.* [23]. Their approach provides a 64% decrease in the detection time over the classic Regular Singular, which makes it suitable for VoIP. Moreover, experimental results prove that this solution is able to detect up to five simultaneous VoIP covert channels with a 100% success rate. Huang *et al.* also introduced the steganalysis method for compressed VoIP speech that is based on second order statistics [24]. In order to estimate the length of the hidden message, the authors proposed to embed hidden data into a sampled speech at a fixed embedding rate, followed by embedding other information at a different level of data embedding. Experimental results showed that this solution not only allows the detection of hidden data embedded in a compressed VoIP call but also to accurately estimate its size.

Hanspach and Goetz present another acoustic covert channel that uses near-ultrasonic sound from a laptop speaker to another laptop's microphone to transfer data [25]. In their work, the acoustic signals themselves shall be unnoticed, while our research addresses hiding of information in digitized acoustic signals, which themselves are not hidden.

In general, steganographic traffic with micro protocols and without micro protocols can both be detected the same way. However, there is work on micro protocol-specific countermeasures that was published by Kaur *et al.* [26]. The authors demonstrate different active and passive attacks on two micro protocols, including attacks on dynamic overlay routing and sniffing attacks. Especially, if designed naively, micro protocols can impact covertness negatively.

2.2. Micro protocols and related work

Micro protocols (also *covert channel control protocols* or *steganographic control protocols*) are protocols featuring headers, which are placed into a steganographic carrier. In other words, they are part of the hidden data transferred in a steganographic transmission and share the available space with the hidden payload of the transmission.

A number of micro protocols exist, which are surveyed and compared in [2]. The provided features of micro protocols differ highly; for instance, [27] presents a micro protocol with application layer functionality that provides a file transfer service, while [3] presents a dynamic routing protocol for steganographic overlay networks. Other common features are to provide reliability for the transfer of steganographic payload or to alternate between different hiding methods on demand to overcome filtering technology in censorship environments.

As micro protocols share the usually limited space within a steganographic carrier with the actual payload, two requirements arise [2]:

(1) The micro protocol header must be as small as possible so that more payload can be transferred per packet as the covertness of a connection increases when less hidden data must be transferred.
(2) The micro protocol header must be embedded into the steganographic carrier in a way that its placement is always conforming to the carrier. Otherwise, the micro protocol may cause unusual states in the carrier (e.g., unusual flag combinations of headers), resulting in lower covertness.

Only two micro protocol engineering approaches exist, and both focus on different goals. Backs *et al.* address the first requirement of a minimized header size in [3]. For this reason, they introduce a dynamic micro protocol header based on the concept of *status updates*. Each header component of the micro protocol is only embedded on demand to reduce the overall size. If a header component indicating a 'state', such as the destination of the attached payload, does not change, the header component indicating the state is not required to be sent again until its state changes. In other words, a status update is performed only on demand. Moreover, the micro protocol header can be fragmented into an arbitrary number of pieces.

The second approach, which consists of six steps, was presented by Wendzel and Keller in [4] and addresses the second requirement; it maximizes the covertness of the micro protocol's placement and behavior. Their approach focuses solely on structured carriers, namely network protocol headers. The ocurrence rates of bits in the micro protocol and bits of the carrier are mapped in a way that the placement of the micro protocol causes as few anomalies as possible. In [2], the approach was optimized in order to reflect bit mappings based on protocol states. For instance, bit mappings of a micro protocol can depend on the state of a Transmission Control Protocol (TCP) connection if embedded into TCP.

Moreover, the carrier and the micro protocol can each be described in form of a formal grammar [4]. Using a language inclusion test, Wendzel and Keller verify whether the micro protocol's design is conforming to the design of the carrier. For example, some bits of carrier headers can only occur at specific phases of a connection and not at the same time as other bits. If the micro protocol violates such rules, the resulting anormal carrier behavior would rise suspicion.

The only micro protocol that partly embeds into unstructured network data was presented by Mazurczyk and Kotulski in [28] but embeds major information into a structured part of the Real-time Transport Protocol (RTP) header and provides no micro protocol engineering approach.

Thus, while Backs *et al.* reduce the size of the micro protocol, Wendzel and Keller optimize the design and embedding of the protocol for structured carriers. No approach is available for the optimal design and placement of micro protocols into unstructured carriers, which is subject of this work.

3. REQUIREMENTS ANALYSIS

In this section, we will first introduce the requirements for the embedding of a micro protocol into audio streams. We describe the implementation details in the following section. Based on our evaluation, we will conduct the actual micro protocol engineering approach for unstructured carriers. The implementation and evaluation are a necessity as no evaluation for the embedding of different micro protocol designs in unstructured carriers exists.

Basic requirements for the embedding of a micro protocol are the non-functional requirements of maximized covertness, robustness against changes, high channel capacity, and a low overhead of the steganographic transmission.

We achieve a high covertness by a space-efficient header design: the fewer bits must be hidden, the smaller the influence on the utilized carrier. Therefore, our micro protocol engineering approach must result in a space-optimized protocol design. For this reason, we apply the concept of the aforementioned status updates.

For the receiver, it must be possible to extract the hidden information of network packets at any time. Two different ways exist to embed a micro protocol into an unstructured carrier:

(1) *Fixed embedding:* the micro protocol header is placed at a fixed position within the audio data.
(2) *Dynamic embedding:* the first bits of the micro protocol header are placed at a fixed position within the audio data. These first bits indicate the placement of the following micro protocol header components within the same packet. In this scenario, the micro protocol header can be split into abitrary parts.

In both cases, the receiver must know the position and length of (the first part of) the embedded micro protocol header. We decided to implement both approaches to enable their comparison. Moreover, the micro protocol receiver must be capable to detect whether a micro protocol is present in a received network packet, or not. For this reason, our approach requires the continuous sending

Table I. Structure of our static micro protocol header.

Attribute	Bits	Comment	Value(s)	Request	Data	Response	Dummy
HT header type	2	**REQ** request	00	●			
		DAT data	01		●		
		RES response	10			●	
		DMY dummy	11				●
NHO next header offset	5		00000...11111	●	●	●	●
FMT data format	1	textual data	0	●			
		binary data	1	●			
CNT packet count	6		000000...111111	●			
VER protocol version	2		01–11	●			
LEN data length	8		0000...1111		●		
CMD command	2	OK	00			●	
		RESEND	01			●	
DMY dummy data	9	OK	9 random bits				●

of dummy data in every network packet to indicate the presence of a micro protocol.

We have chosen the simple LSB algorithm, which hides data in LSBs, for the implementation of our proof of concept system. We used LSB as our work does not focus on the hiding algorithm of the covert data but on the design and placement of the micro protocol itself. Hence, in a real implementation, an algorithm would be used that is better suited against steganalysis but provides a similar quality, that is, does not detoriate the audio quality notably.

4. IMPLEMENTATION

This section describes the micro protocol header design. The design is implemented in our proof of concept system and is evaluated in the following section.

The design of a micro protocol header can be performed in two ways. Either, the whole micro protocol header is transferred within a single network packet (refered to as a *static* design). Or, the micro protocol is split over multiple packets were the earlier packets indicate that micro protocol header parts follow with the next packet (refered to as *dynamic* design). We implement both design approaches in order to compare them and implemented them additionally using fixed embedding (see previous section). The combination of these design and embedding approaches allows a highly dynamic header structure and embedding.

We decided to implement a status update-based micro protocol header consisting of *attributes*. Each micro protocol header starts with an attribute called *header type* (HT) that defines the structure of the remaining header.

4.1. Design of a static micro protocol header

Static headers comprise the advantage of a clear structure. On the other hand, static headers cannot be split over different network packets, and they may comprise header components, which are not required in every transferred packet. We defined the following four HTs for a generic, static micro protocol header:

(1) REQ: a packet that initiates a request (incl. following data packets of a request),
(2) DAT: a data packet containing payload,
(3) RES: a response packet, containing no data as, in our model, responses only indicate whether a request was successful (ACK) or not (ERR), and
(4) DMY: a dummy type to indicate the presence of a micro protocol without content.

The HT is followed by a number of other attributes as shown in Table I that also defines which attribute is included in each HT. The attributes are defined as follows:

- NHO: an offset indicating the start of the next header attribute within the following packet,
- FMT: specification of the data format in a request (ASCII plaintext or binary data),[†]
- CNT: counter for data packets (DAT HT),
- VER: version of the micro protocol used (only dynamic header),
- LEN: the length of attached data (only dynamic header),
- CMD: specifies a command (‘OK’ or ‘RESEND’) to indicate the need of a re-transmission, and
- DMY: indicating 9 bits of dummy data.

Figure 1 visualizes a sample packet for a static micro protocol header that is of the HT ‘request’. The whole header is embedded into the audio data transferred over the network. The HT starts at a fixed position (fixed embedding) and indicates a request (REQ), which is followed by the next header offset, the format bit, and the micro protocol’s version number.

[†] Responses do not contain payload (DAT) but the bi-directional communication allows every peer to transfer REQ packets containing payload.

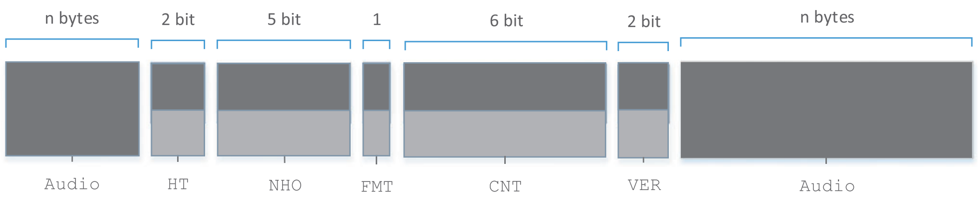

Figure 1. Example for a request using our static micro protocol header.

Table II. Structure of our dynamic micro protocol header.

Attribute	Bits	Description	HT value	Request	Response	Dummy
HT header type	3	**REQ** request	000	•		
		RES response	001		•	
		DMY dummy	010			•
		DAT data	011	•		
		LEN data length	100	•		
		NHO next header offset (optional)	101	•		
		FMT data format (optional)	110	•		
		VER protocol version (optional)	111	•		
VAL value	$1 \dots n$	allows to transfer payload		•	•	•

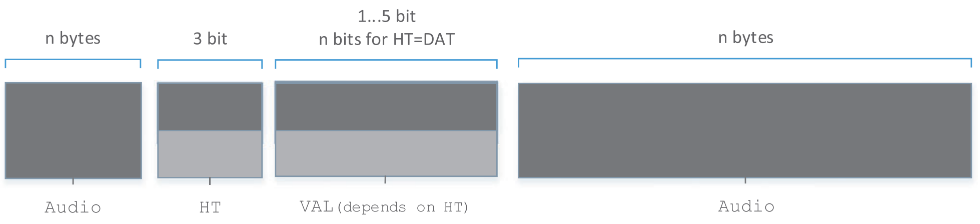

Figure 2. Visualization of the general structure of a dynamic micro protocol header. Each packet contains only one header type (HT), that is, no 'pointer' to the start of next HT is required.

4.2. Design of a dynamic micro protocol header

Dynamic header designs allow to split headers into an abitrary number of data chunks. These data chunks can be embedded at different locations of an audio payload. While dynamic header designs result in higher implementation cost, they also require the re-structuring of our previously introduced static header. We apply the same attribute types as for the static header, but most of these attribute types are now HTs (Table II).

We distinguish between primary and secondary HTs. Primary HTs indicate the overall meaning of a header (request, response, or dummy data) while secondary HTs depend on the primary HTs. Secondary HTs add additional attributes to the primary HT. In other words, secondary HTs fullfil the same role as attributes in the *static* header design, but they can now be *included on demand* and split over multiple locations within the audio carrier (by using their HT value to indicate their presence).

All secondary HTs are only included if a request is sent, which results in smaller response and dummy messages. Requests are additionally shrinked in size by defining rarely used secondary HTs as optional (indicated by the HT; Table II). In comparison to the static header, DAT was replaced by a value field (VAL) of dynamic size that can indicate both, payload and the parameters for header fields, such as a length value for LEN or a version number for VER. The general structure of our dynamic header design is shown in Figure 2.

The start and the end of a transmission is signaled via request headers as *Begin-of-Message* and *End-of-Message* headers. The former is indicated by $HT = REQ$, $VAL = 0$, and the latter is indicated by $HT = REQ$, $VAL = 1$.

We visually compare both design concepts, the static and the dynamic header, in Figure 3.

5. EVALUATION

We now evaluate the implementation of both the static and the dynamic micro protocol designs. These results will serve as a basis for the design of our micro protocol engineering approach for unstructured carriers.

For evaluation purposes, we chose to emulate an audio streaming service like VoIP by streaming a predetermined

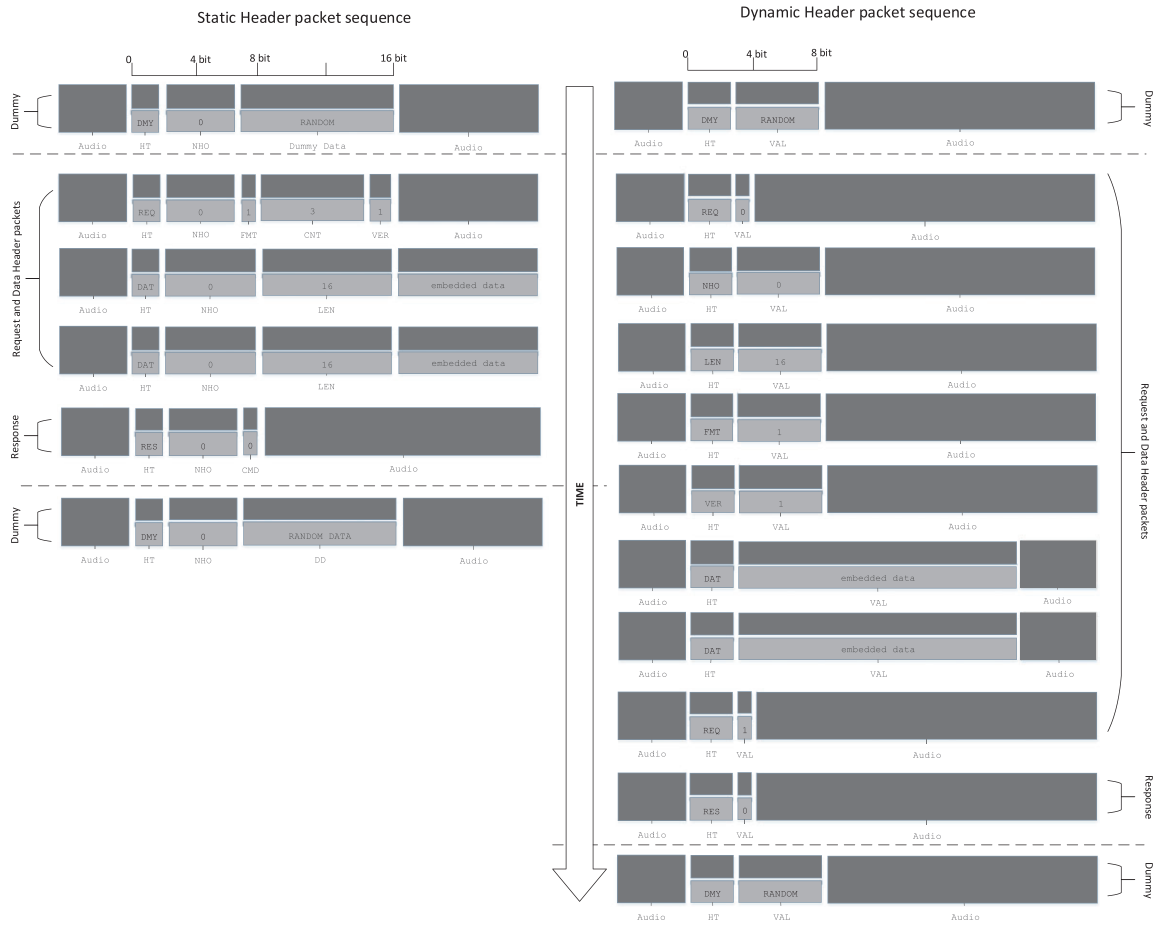

Figure 3. Comparison of static and dynamic micro protocol header designs.

5-min long .wav file. It contains audio recordings from the TIMIT [29] continuous speech corpus, which is one of the most widely used corpora in the speech recognition community. Both male and female voices speaking English were used. During the experiments, parts of the .wav input file were then inserted into the payloads of consecutive RTP packets and sent to the receiver where reassembling into a .wav file was performed. Then, the original and degraded files were compared with the use of *Perceptual Evaluation of Speech Quality* [30], and the resultant Mean Opinion Squared-Listening Quality Objective (*MOS-LQO*) was returned.

We compared the transmission of hidden data using both micro protocol designs with the same payload sizes for different variations of the LSB algorithm and different audio codecs. For both the static and the dynamic header designs, we performed transmissions with varying header structures and varying frequencies at which packets were sent. For this reason, we implemented three scenarios:

(1) A transmission of dummy packets with 16 bit payload.
(2) A transmission using a packet loop, each consisting of dummy packets, one REQ, few DAT, and one RES packet. We sent requests with 3×480 bit payload. As shown in Figure 3, it was necessary to add additional packets to the dynamic header, namely for the HTs NHO, LEN, FMT, and VER.
(3) A transmission using one REQ packet followed by as many DAT packets as feasible per request and header design.

All requests were acknowledged with response packets in order to perform a bi-directional transmission. In detail, each test scenario works as follows: First, the original pulse code modulation file will be read out. Second, the transmitter converts the file to the defined payload type, and the transmitted bytes are recorded as raw bytes into a separate file. Third, the receiver reads the stream and decodes the embedded message if included. Fourth, the stream is saved in a .wav file. Fifth, streams with embedded messages and streams without embedded messages will be compared regarding changes at the bit level and their audio quality.

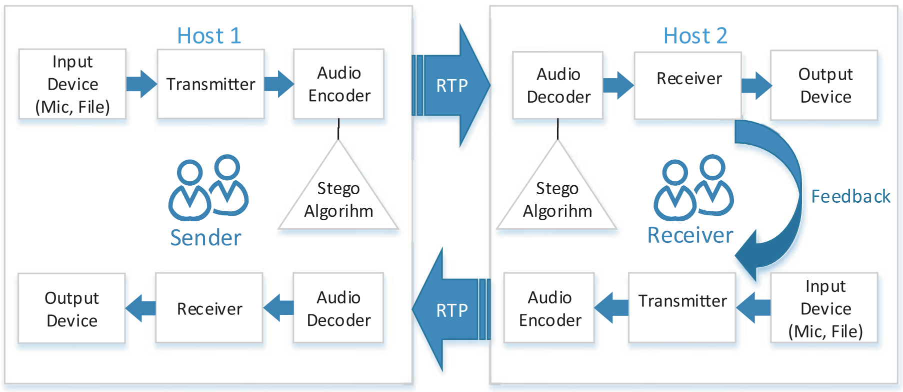

Figure 4. Experimental setup for the evaluation of our micro protocol designs.

Table III. Evaluation results for a static micro protocol header design.

Codec	Algorithm	Hidden Bits	MSE	SNR (dB)	PSNR (dB)	MOS-LQO
ULAW	none (reference)	0%				4.25
(8 Khz, 8 bit,	LSB1	0.417%	0.017	63.881	65.861	4.080
mono)		3.853%	0.155	54,240	56.220	3.285
		11.847%	0.478	48.357	51.339	2.577
	LSB2	0.417%	0.057	58.554	60.543	4.099
		7.219%	0.918	46.509	48.505	3.054
		23.094%	2.952	41.398	43.429	2.720
	MSB	0.417%	289.717	21.490	23.511	2.801
		3.853%	2581.209	11.714	14.013	1.687
		11.847%	7914.764	6.149	9.146	1.332
	LSB6Enh	0.417%	15484	34.239	36.232	3.501
		3.845%	135.814	24.713	26.801	2.938
		11.847%	412.661	19.661	21.975	2.523
DVI	none (reference)	0%				3.85
(11 Khz, 4 bit,	LSB1	0.410%	0.016	59.927	66.226	3.847
mono)		4.123%	0.163	49.697	55.996	2.978
		11.833%	0.473	45.082	51.382	2.678
	LSB2	0.820%	0.071	53.318	59.617	3.101
		6.213%	0.517	44.686	50.995	1.951
		18.799%	1.548	39.899	46.232	1.457

We compare the Mean Squared Error (*MSE*), Signal-to-Noise Ratio (*SNR*), Peak SNR (*PSNR*), and MOS-LQO of our proof-of-concept implementation to evaluate the differences between audio transmissions without steganographic content and those with steganographic content.

Figure 4 shows our experimental setup. Two hosts are communicating over a covert channel, and the receiver provides feedback about the received steganographic data back to the sender. We applied the audio codecs ULAW and DVI; all transmissions are performed using the RTP protocol.

5.1. Static header design

The results for the evaluation of our static header design are shown in Table III, which shows the MSE, SNR, PSNR, and MOS-LQO values for different codecs and algorithms. Increasing the amount of embedded data results in a lower covertness, in a lower MOS and needs fewer packets with embedded payload per transaction. Table III also reveals the insufficient performance for the MSB and LSB6Enh algorithms (the latter is utilizing the sixth LSB). We can conclude that changes of higher significant bits result, as expected, in a higher reduction of the voice quality when a static micro protocol is embedded. Figure 5 additionally visualizes the SNR, PSNR, and MOS values for the static micro protocol header case depending on the embedding algorithm used, codec and utilized embedding capacity.

5.2. Dynamic header design

The results for our experiments using the dynamic header are shown in Table IV. Because of the additional HT, the dynamic header consumes more space than the static

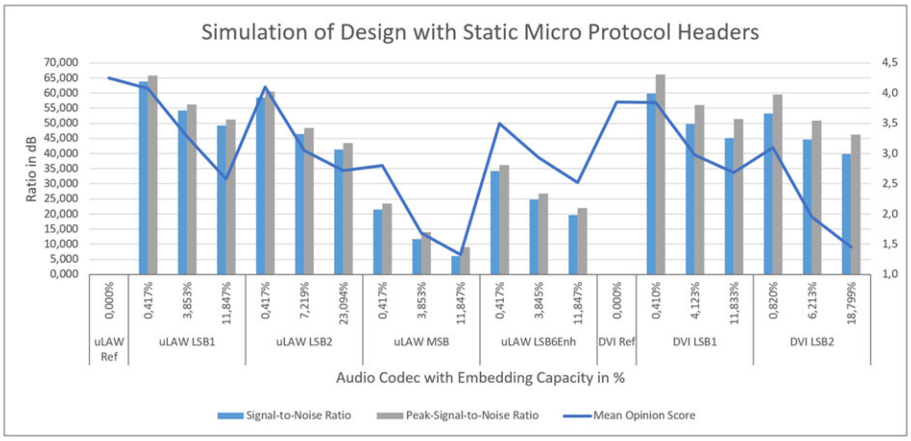

Figure 5. Comparison of experimental results for the static micro protocol design.

Table IV. Evaluation results for a dynamic micro protocol header design.

Codec	Algorithm	Hidden Bits	MSE	SNR (dB)	PSNR (dB)	MOS-LQO
ULAW	none (reference)	0%				4.25
(8 Khz, 8 bit, mono)	LSB1	0.208%	0.008	66.896	68.875	4.242
		2.457%	0.100	56,133	58.113	3.429
		11.549%	0.469	49.437	51.420	2.636
	LSB2	0.208%	0.025	62.234	64.214	4.248
		4.704%	0.602	48.348	50.338	3.290
		22.785%	2.915	41.454	43.484	2.722
	MSB	0.208%	143.555	24.562	26.561	3.327
		2.457%	1674.369	13.712	15.892	1.946
		11.549%	7777.446	6.244	9.222	1.343
	LSB6Enh	0.208%	7.583	37.347	39.333	3.805
		2.457%	87.970	26.639	28.687	3.092
		11.549%	405.232	19.746	22.054	2.537
DVI	none (reference)	0%				3.85
(11 Khz, 4 bit, mono)	LSB1	0.410%	0.016	59.914	66.213	3.847
		1.690%	0.069	53.475	59.773	3.567
		6.918%	0.280	47.362	53.661	3.302
	LSB2	0.410%	0.029	57.264	63.562	3.439
		2.526%	0.209	48.627	54.930	2.885
		11.093%	0.918	42.185	48.504	2.494

header in case of small payload transfers. In other words, our highly fragmented header design produces an overhead of header data in comparison with the static header. For this reason, the dynamic header leaves slightly less space for the embedding of hidden payload[‡] and the average embedding capacity for our test cases was higher in case of static headers (6.383%) as for dynamic headers (5.188%). However, if large payload is transferred, the overhead of the dynamic header is smaller than the overhead of the static header and thus provides better capacity. The MOS results are better in case of the dynamic header (in average 3.119 for dynamic header and 2.812 for the static header).

Figure 6 visualizes the SNR, PSNR, and MOS values for the dynamic micro protocol header depending on the embedding algorithm, codec and utilized embedding capacity.

6. MICRO PROTOCOL ENGINEERING

Based on the results conducted in the previous experiments, we now derive recommendations for the design of micro protocols in unstructured carriers. We afterwards present our micro protocol engineering approach.

[‡] Backs *et al.* provide another dynamic header approach that provides slightly better results than a static header [3]. Their header design provides only few fragments and thus decreases the overhead in comparison to a static header.

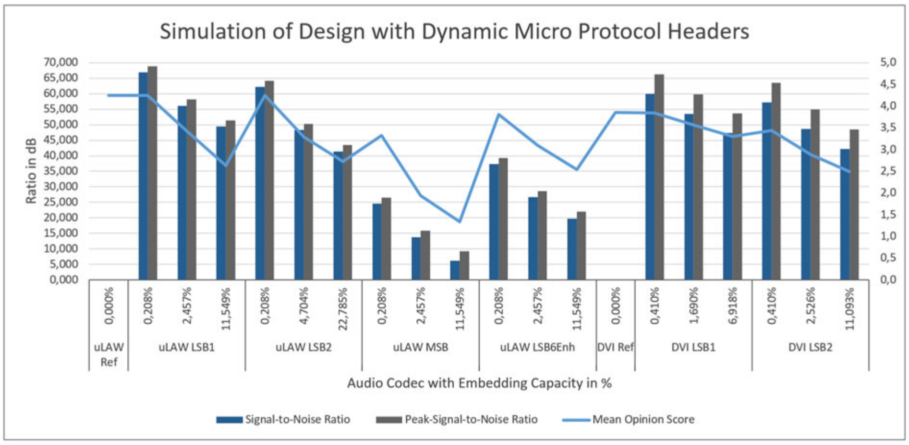

Figure 6. Comparison of experimental results for the dynamic micro protocol design.

6.1. General recommendations

Even small changes in audio transmissions can lead to the acoustic identification of quality reduction.[§] At the same time, the embedding of larger amounts of data per packet without raising attention is difficult in highly compressed audio transmissions. A first recommendation is therefore that implementations of micro protocols should allow the selection of particular codecs on start-up, depending on the amount of data to be transferred and other considerations such as time limits.

Our following recommendations are related to requirements, namely covertness, channel capacity, and reliability. Indeed, these requirements result in a multi-criteria optimization problem, as a higher channel capacity results in lower covertness and may also influence reliability.

6.1.1. Maximizing covertness.

An alternating offset for the initial 'HT' field of the micro protocol header increases the hurdles for a warden to recognize patterns of steganographic transmissions and with it the hidden communication itself. We also recommend the introduction of dummy data packets (as forseen in our protocol designs). These dummy data are exchanged between hosts when no actual steganographic payload must be transferred. If no dummy data are transferred, the difference between actual steganographic transfers and audio transfers without steganographic content will be larger and thus may rise more suspicion. A small micro protocol header contributes additionally to the covertness and must be seen as a central element in any micro protocol engineering approach. We recommend the implementation of *dynamic* headers as these result in a higher covertness.

[§] Regarding to Dickreiter *et al.*, humans are able to recognize a voice inside a noised signal with a SNR of 6 dB [31].

6.1.2. Maximizing capacity.

Codecs with lower compression allow the integration of more hidden data. Like in case of a maximized covertness, a small micro protocol header can contribute to a maximized capacity, too: the fewer space is required for embedding the header, the more space is left for embedding hidden payload. Advanced techniques for the embedding of audio steganography, such as Transcoding Steganography (TranSteg) [32], can be applied to increase the channel capacity. Moreover, a covert channel can utilize structured and non-structured elements of network packets simultaneously. For larger payloads, the covertness of the dynamic header provides better results and is recommended for this reason. For small payload transfers, the overhead of the dynamic header results in lower capacity than the static header approach. For this reason, the protocol engineer must decide — based on the particular use case of the micro protocol — which approach to chose if the capacity must be optimized.

6.1.3. Reliability.

In case of real-time streaming, the loss of audio packets occurs on a regular basis. If audio packets are lost, their steganographic content is lost as well. For this reason, we recommend the implementation of reliability mechanisms, such as Automatic Repeat reQuest or more advanced techniques. It is also imaginable to implement reliability not on a per-packet basis but instead on a per-*n*-packet basis, for example, for every 10 packets. This heavily reduces the required number of acknowledgement messages.

Of course, knowledge about the reliability mechanism used can also be exploited to detect that the covert communication is taking place, for example, by intentionally introducing packet losses in order to observe and to correlate the responses. In such a case, a more sophisticated reliability approach can be utilized as proposed, for example, by Hamdaqa and Tahvildari [33], which can be easily incorporated for audio steganography purposes. It provides a reliability and fault tolerance mechanism based on a modified (k, n) threshold using Lagrange Interpolation, and their results demonstrate that the complexity of steganalysis is increased.

6.2. Micro protocol engineering approach

While Section 3 discussed the requirements of our micro protocol implementation, we now present the requirements of our micro protocol engineering approach. The micro protocol engineering approach is based on a typical use case for which a micro protocol design will be created. The output of the micro protocol engineering process is the micro protocol design.

In [4], Wendzel and Keller mention fundamental requirements of a micro protocol engineering approach, of which we consider the following two for our approach:

- The micro protocol engineering approach must be applicable in practice.
- The most important criterion to fulfill by the micro protocol is to provide a high covertness. Therefore, the protocol engineering approach must support a maximized covertness of the resulting micro protocol.

We extend these aspects with the following additional requirements:

- The micro protocol's transmission must be reliable, that is, possess a feedback channel or another means for error correction.
- The micro protocol should be adaptable to different codecs used for the unstructured carrier.

Based on these requirements, we derive a four-phase model for the micro protocol designing process. We limit the required workload of the protocol engineer and design the particular steps in a practically applicable manner. The model is shown in Figure 7. A switch between the phases is feasible at any time to optimize the micro protocol design.

6.2.1. First phase.

In the first phase, a use case analysis must be performed. The use case analysis defines the requirements (e.g., high channel capacity), which are linked to that use case. For instance, some micro protocols must be designed for high-throughput transmissions while others may only need to transfer a few bits per second. A sample use case could be the transfer of illicit video data over slow or non-reliable connections. Moreover, it must be defined whether a real-time transmission or the transmission of data on-demand will be required and which transmission media will be used. Finally, it must be defined whether the steganographic transmission must be reliable.

6.2.2. Second phase.

Based on the requirements definition in the first phase, the second phase defines the network protocols, the media codecs, the compression algorithm used, and the steganographic embedding algorithms to be used. For instance, if the requirement is a real-time transmission, the RTP protocol can serve as a network protocol.

If steganographic data is embedded into the audio data *after* the compression took place, the hidden data

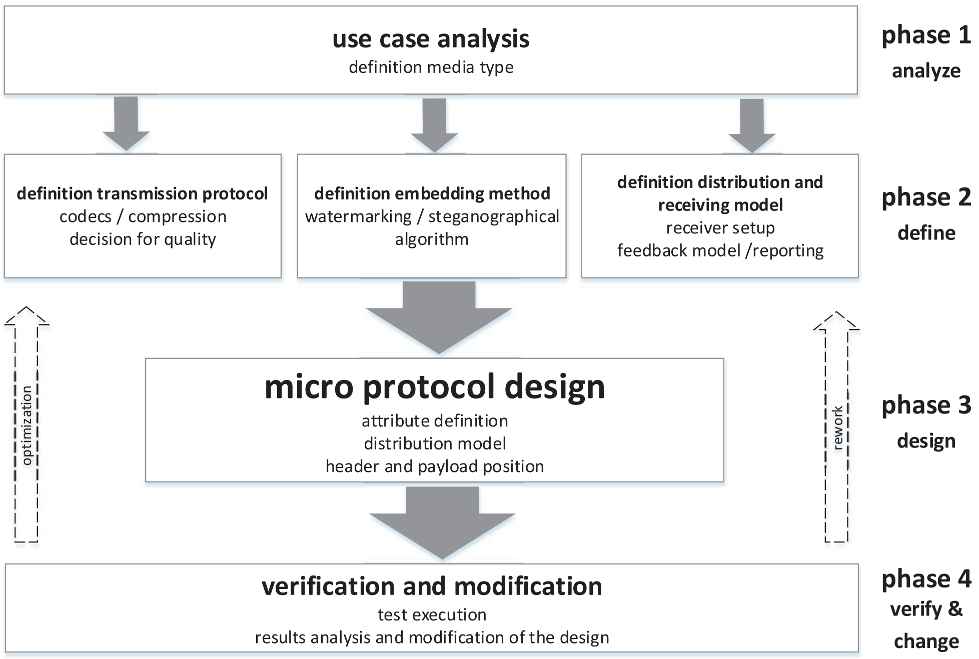
use case analysis
definition media type
phase 1
analyze
definition transmission protocol
codecs / compression
decision for quality
definition embedding method
watermarking / steganographical algorithm
definition distribution and receiving model
receiver setup
feedback model /reporting
phase 2
define
optimization
micro protocol design
attribute definition
distribution model
header and payload position
rework
phase 3
design
verification and modification
test execution
results analysis and modification of the design
phase 4
verify & change

Figure 7. Our phase model for the design of micro protocols in unstructured carriers.

may not be subjected to change during transmission. In such a case, *spatial-domain* methods such as LSBs or adding information to silence sections of the voice transmission can be applied. If changes of the steganographic content can be performed within the transmission, it may be more suitable to select *frequency-based* methods such as Phase Coding, Spread Spectrum, or Echo Data Hiding.

Finally, if reliability was defined as a requirement within the first phase, a means must be defined to realize reliability, for example, by adding sequence numbers and a TCP-like reliability algorithm or more covert approaches like described in Section 6.1.

6.2.3. Third phase.

First, the protocol engineer must determine the available space to place hidden data into the unstructured carrier. The available space is a result of the audio codec used. The resulting value is used as a limit for the size of the micro protocol header.

Secondly, the requirements for covertness and channel capacity must be addressed. As discussed in Section 6.1, dynamic header designs can increase the covertness and — for larger payload — the capacity of a transmission while for small payload transactions, a static header will provide a better capacity. After this decission is made, the required size for the micro protocol header must be determined. All necessary features, such as the capability to transfer dummy data or to provide reliability, must be integrated into the header (represented by single or multiple bits as header fields). If the micro protocol consumes too much space, features must either be reduced or the micro protocol must be designed in a more dynamic way. If both actions are not feasible, a carrier that provides more space must be selected, which leads back to the second phase.

Thirdly, the protocol engineer subtracts the micro protocol header size from the available hidden storage space within the carrier. The result is the available space for the hidden payload per packet. If the available space is insufficient to transfer the necessary number of bits per second in the given use case, the engineer must redesign the micro protocol or must change definitions within the second phase.

Fourthly, the protocol engineer must specifiy how the start of micro protocol data within the carrier is indicated. This can be carried out by defining a fixed offset for the first packet and defining sequences to identify data.[¶]

Fifthly, the micro protocol must be implemented in a proof of concept system.

6.2.4. Fourth phase.

Finally, the micro protocol design must be verified. For this reason, a proof of concept code must be implemented and tested. Passive wardens, either as humans or as software, should be used to assess the non-detectability of the covert communication. Moreover, it must be evaluated whether the embedding of the steganographic data can result in behavior that is not conforming to a codec's specification. This last step includes ensuring that attributes defined in the audio codec (e.g., the packet capacity, specified sampling rate, bit depth, or needed audio transformation steps) are not violated by the embedded micro protocol. A simple method is to verify the correct processing of the modified streams by a common player software, for example, a streaming client.

[¶] The larger the allowed next header offset values (header type NHO), the less space will be available per packet (see third step).

7. CONCLUSIONS AND FUTURE WORK

We demonstrated the embedding of a micro protocol into an unstructured carrier, namely, audio streaming over the network. The micro protocol was implemented in a static and a dynamic way to compare the resulting performance and impact of both designs on the audio signal. Based on this feasibility study and its evaluation, we developed a micro protocol engineering approach for unstructured carriers. Our micro protocol engineering approach extends the existing approaches, which were solely focusing on structured carriers.

Micro protocols can be designed in a different way as proposed in this article, for example, more header functionality can be integrated and alternative audio codecs can be applied. However, our discussed designs are highly dynamic and, for this reason, represent a generic design approach that can easily be modified by introducing new or replacing present HTs.

At the moment, our framework does not address the micro protocol's temporal behavior. However, the micro protocol's temporal behavior should at least be adjusted closely to the temporal behavior of the carrier (e.g. inter packet gaps) — otherwise its temporal behavior may lead to detectability [26]. The micro protocol engineering framework will thus be extended for this purpose.

REFERENCES

1. Zander S, Armitage G, Branch P. A survey of covert channels and countermeasures in computer network protocols. *IEEE Communications Surveys and Tutorials* 2007; **9**(3): 44–57.
2. Wendzel S, Keller J. A survey and outlook on covert channel-internal control protocols. *Annals of Telecommunications* 2014; **69**(7): 417–430.
3. Backs P, Wendzel S, Keller J. Dynamic routing in covert channel overlays based on control protocols, *Proceedings of the International Workshop on Information Security, Theory and Practice (ISTP-2012)*, IEEE, London, United Kingdom, 2012; 32–39.

4. Wendzel S, Keller J. Systematic engineering of control protocols for covert channels, *13th Joint IFIP TC6 and TC11 Conference on Communications and Multimedia Security (CMS 2012)*, Canterbury, United Kingdom, 2012; 131–144.
5. Zielińska E, Mazurczyk W, Szczypiorski K. Trends in steganography. *Communications of the ACM* 2014; **57**(3): 86–95.
6. Bender W, Gruhl D, Morimoto N, Lu A. Techniques for data hiding. *IBM Systems Journal* 1996; **35**(3 and 4): 313–336.
7. Katzenbeisser S, Petticolas F. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House: Norwood, MA, United States, 2000.
8. Kirovski D, Malvar HS. Spread-spectrum watermarking of audio signals. *IEEE Transactions on Signal Processing* 2003; **51**(4): 1020–1033.
9. Fisk G, Fisk M, Papadopoulos C, Neil J. Eliminating steganography in internet traffic with active wardens, *Proceedings Information Hiding*, Springer Berlin Heidelberg, Noordwijkerhout, The Netherlands, 2003; 18–35.
10. Aoki N. A packet loss concealment technique for VoIP using steganography, *Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2003)*, Chiang Mai, Thailand, 2003; 470–473.
11. Dittmann J, Hesse D, Hillert R. Steganography and steganalysis in voice-over IP scenarios: operational aspects and first experiences with a new steganalysis tool set, *Proc of SPIE 2005, Vol. 5681, Security, Steganography, and Watermarking of Multimedia Contents VII*, San Jose, CA, United States, 2005; 607–618.
12. Wu Z, Yang W. G.711-based adaptive speech information hiding approach. In *Intelligent Computing*, vol. 4113, Huang DS, Li K, Irwin G (eds), Lecture Notes in Computer Science. Springer Berlin Heidelberg: Kunming, China, 2006; 1139–1144.
13. Wang C, Wu Q. Information hiding in real-time VoIP streams, *Multimedia, 2007. ISM 2007. Ninth IEEE International Symposium on*, Taichung, China, 2007; 255–262.
14. Takahashi T, Lee W. An assessment of VoIP covert channel threats, *Security and privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, Nice, France, 2007; 371–380.
15. Liu L, Li M, Li Q, Liang Y. Perceptually transparent information hiding in G.729 bitstream, *Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP '08 International Conference on*, Harbin, China, 2008; 406–409.
16. Tian H, Zhou K, Huang Y, Feng D, Liu J. A covert communication model based on least significant bits steganography in Voice over IP, *Young Computer Scientists, 2008. ICYCS 2008. the 9th International Conference for*, Hunan, China, 2008; 647–652.
17. Tian H, Zhou K, Jiang H, Liu J, Huang Y, Feng D. An m-sequence based steganography model for Voice over IP, *Communications, 2009. ICC '09. IEEE International Conference on*, Dresden, Germany, 2009; 1–5.
18. Xu T, Yang Z. Simple and effective speech steganography in G.723.1 low-rate codes, *Wireless Communications Signal Processing, 2009. WCSP 2009. International Conference on*, Nanjing, Germany, 2009; 1–4.
19. Tian H, Jiang H, Zhou K, Feng D. Transparency-orientated encoding strategies for Voice over IP steganography. *The Computer Journal* 2011, http://comjnl.oxfordjournals.org/content/early/2011/11/03/comjnl.bxr111.abstract.
20. Xu E, Liu B, Xu L, Wei Z, Zhao B, Su J. Adaptive VoIP steganography for information hiding within network audio streams, *Network-Based Information Systems (NBiS), 2011 14th International Conference on*, Tirana, Albania, 2011; 612–617.
21. Liu J, Zhou K, Tian H. Least-significant-digit steganography in low bitrate speech, *Communications (ICC), 2012 IEEE International Conference on*, Ottawa, ON, Canada, 2012; 1133–1137.
22. Kraetzer C, Dittmann J. Mel-cepstrum based steganalysis for VoIP steganography, *SPIE Proceedings Vol. 6505: Security, Steganography, and Watermarking of Multimedia Contents IX*, San Jose, CA, United States, 2007; 650505-1–650505-12.
23. Huang YF, Tang S, Zhang Y. Detection of covert voice-over internet protocol communications using sliding window-based steganalysis. *Communications, IET* 2011; **5**(7): 929–936.
24. Huang Y, Tang S, Bao C, Yip YJ. Steganalysis of compressed speech to detect covert voice over internet protocol channels. *Information Security, IET* 2011; **5** (1): 26–32.
25. Hanspach M, Goetz M. On covert acoustical mesh networks in air. *Journal of Communications* 2013; **8**(11): 758–767.
26. Kaur J, Wendzel S, Meier M. Countermeasures for covert channel-internal control protocols, *Proceedings 4th International Workshop on Cyber Crime (IWCC)*, IEEE, Toulouse, France, 2015; 422–428.
27. Trabelsi Z, Jawhar I. Covert file transfer protocol based on the IP record route option. *Journal of Information Assurance and Security (JIAS)* 2010; **5**(1): 64–73.
28. Mazurczyk W, Kotulski Z. New security and control protocol for VoIP based on steganography and digital

watermarking. *Annales UMCS, Informatica* 2006; **AI 5**: 417–426.

29. Garofolo JS. *et al.*. *TIMIT acoustic-phonetic continuous speech corpus linguistic data consortium*, 1993.
30. International TU. ITU-T recommendation P.862, perceptual evaluation of speech quality (pesq): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs, 2001.
31. Dickreiter M, Dittel V, Hoeg W, Wöhr W. *Handbuch der Tonstudiotechnik*, 7th edn. De Gruyter Saur: Berlin, Germany, 2008. (in German).
32. Mazurczyk W, Szaga P, Szczypiorski K. Using transcoding for hidden communication in IP telephony. *Multimedia Tools and Applications* 2014; **70**(3): 2139–2165.
33. Hamdaqa M, Tahvildari L. ReLACK: A reliable VoIP steganography approach, *Secure Software Integration and Reliability Improvement (SSIRI), 2011 Fifth International Conference on*, Jeju Island, South Korea, 2011; 189–197.