A Framework for Synthetic Stego*

[Extended Abstract]

Philip Ritchey†
Dept. of Computer Sciences
Purdue University
IN 47906
pritchey@cs.purdue.edu

Jorge R. Ramos‡
Dept. of Computer Sciences
Purdue University
IN 47906
ramos@cs.purdue.edu

Vernon Rego
Dept. of Computer Sciences
Purdue University
IN 47906
rego@cs.purdue.edu

ABSTRACT

We present a technique for hiding information in stochastic settings via data-synthesizing schemes based on transform-expand-sample (TES) processes. The technique is applicable whenever data generated by an application or process is sufficiently complex to exhibit random but structured behavior (such as in collective data transforms), and data trajectories have viable alternatives that are unverifiable or simply hard to verify. In such cases, a synthesizing procedure generates novel data that either actually replaces, or is generated instead of, application or process data. When information can be hidden in such data at levels higher than typical levels of noise, message-neutralizing attacks will fail; and if synthetic data, stego data and application/process data cannot be distinguished, secure stego transmissions can be launched. An information-theoretic model shows that such hiding techniques are arbitrarily secure. We present some experimental results.

1. INTRODUCTION

Practical steganography [1] deals with the *embedding* of secret messages in what is ordinarily assumed to be an innocent transfer of information between two agents: *Alice* and *Bob*. The embeddings may be done in a variety of context-dependent ways and, in its simplest form, may not involve additional encryption. They are context-dependent, in general, because they seek to escape the attention of a warden *Wendy*, whose job it is to monitor all communications for potential hidden information. It is in Alice's best interests to exploit the context as much as possible, to embed information that cannot be neutralized by algorithms which obliterate content that may be hiding in noise. In this paper, we briefly present a representative example from new class of steganographic embeddings cast in a stochastic framework. We focus on arbitrary applications or processes that generate random data with some structure, and where Alice may weave herself into generation process. For ease of exposition, we will use the standard steganographic model [2].

Concisely, let $\mathcal{G}$ be some *generator*, i.e., an application or process that generates *base* data that is relevant to consumers who interact with the application/process. If Alice is able to intercept data generated by $\mathcal{G}$, then Alice is free to utilize a clone generator $\mathcal{G}'$, at least for a small period of time, to generate *clone* data which can then function as coverdata for the application/process. If the data source is reasonably complex *and* $\mathcal{G}'$ has strong cloning properties, there will generally be no easy way to discriminate between base data and clone data. Discrimination may be impossible because clone data automatically replaces nonexistent application- or process-data. In such cases, the only meaningful tests will involve historical data, and the problem of discrimination will reduce to one of distinguishing between (practically) identical empirical distributions, or empirical time series — truly challenging problem in statistics.

Given the above setting, let $S = \{X_1, X_2, \cdots, X_n\}$, $n > 0$, be the coverdata generated by $\mathcal{G}'$, where S is a stochastic sequence. It is possible that S is non-stationary, with complex dependencies, though we will assume stationarity. We will also assume that S is public and perhaps also broadly visible, so that Alice may use S as a channel to broadcast secret information. Though the information can be communicated in any form, we use numerical (stochastic) data to demonstrate the idea, simultaneously highlighting the presence of structure in random data for many applications and processes. Thus, Alice works with *base* data and uses $\mathcal{G}'$ to generate *novel* data for message-hiding. We thus have two problems: first, clone data generation (synthesis), and second, message embedding (stego).

With access to a secret-key stegosystem [2], Alice is free

*Research supported by NSF CNS-0716398. CSIIRW '09, April 13-15, Oak Ridge, Tennessee, USA 978-1-60558-518-5 ... 5.00

†Ph.D. Program.

‡Presently at Microsft Corp., Canada.

to utilize the system's embedding methodology within S to obtain a *stego version* S'. Because S is a stochastic process, and because S' relies on S and on several other creative processes for its construction, we call S' a *stego process*. When Alice generates and sends S' to Bob, Wendy observes information in transmission. If the stego process is secure, Wendy will not recognize the transmission as a *stego process transmission* or SPOT. Thus, Alice's task is to convert $\mathcal{G}'$'s output into SPOTS, with the intention of misleading Wendy (who cannot tell $\mathcal{G}$ from $\mathcal{G}'$) and simultaneously transmitting secret information to Bob; and when S' is public, to all of Bob's distributed counterparts as well. In dealing with security bounds we may assume that Wendy has access to the probability structure of $\mathcal{G}'$, or equivalently S. In Section 2 we present a very brief overview of the methodology and explain how TES processes are used to build a generator $\mathcal{G}'$. In practice any of a variety of message embeddings may be used along with TES, as will be shown from experimental results.

2. SYNTHETIC STEGO PROCESSES

Digital stego comes typically in one of three forms [10] with typical cover files being text, image and audio files. In *injection*-stego, a message m is put inside an unaltered cover c, increasing cover size — similar to a Trojan horse [19]. In *substitution*-stego, m replaces noise-level information in c (e.g., rarely used segment in executable code, least significant pixel bits in images); this bit-level manipulation can cause (still or video) image-degradation, poorer audio quality, or errors in executables. In *synthetic*-stego, a new stego object s is synthesized in place of an authentic cover c — music, text, some graphic or image, etc. — using a *mimicking* procedure for duplicity. There are other methods which overlap with the above methods and are not easily classified. These include transform-based, spread-spectrum based, distortion-based and statistics-based methods [10]. Successful cover objects may include executable files [4, 3], link-layer frames and TCP/IP [5, 17], timestamps and ICMP messages [5], and video and audio streams [22]. Image- and text-steganography and their analysis dominate much of the hidden-information landscape.

It appears that steganographic techniques dealing with the the synthesis of coverdata has not receive much attention. In our experience, data generation can play a pivotal role in stego applications simply because data is ubiquitous, verification of the authenticity of arbitrary data streams can be a serious challenge, and data from random sources introduces additional complexity and uncertainty. It is possible that a key reason for overlooking data duplicity stems from the difficulty of creating data covers that resemble other covers in random settings. Given a duplicate cover, the question then becomes one of efficient stego embeddings and ascertaining potential limits to discovery.

Our proposed approach for synthetic steganography based on stochastic data is summarized in Fig.1. Given an authentic data object D_n (e.g., from a data-generating application such as a simulation, any application whose numerous, complex input parameters preclude repetition as an experiment, a sequence of transactions or simply any random process), we construct a stochastic model that can be used to generate a background sequence Y_n with certain properties. This

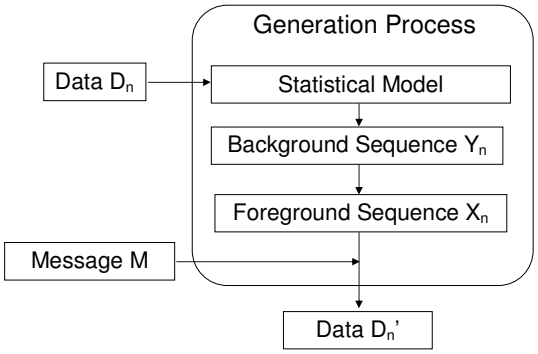

Figure 1: Information Hiding in Stochastic Sequences

background sequence is then transformed into a foreground sequence X_n in a way that preserves key statistical statistical properties of the data object D_n. We have a choice of whether to embed a stego message in the background or in the foreground, to finally obtain a stego object $D_n(\delta)$, for some embedding parameter δ.

2.1 Data Synthesizers

In the following we give a brief review of existing data synthesizers. A useful taxonomy can be found in [12]. Models with independence involve data based on independent observations with no dependence on time. These may include well-known statistical distributions (binomial, normal, exponential, etc.) [11] or represent the output of processes which generate independent samples. There are well-known statistical models for generating consecutive observations that depend on time or have dependency structure, such as ARIMA, Markov-modulated Poisson processes, semi-Markov chain generators etc., which are parametric models that can be used to fit existing data and make predictions.

One approach to synthesize time series is by the use of expressions of the form $y(x) = \nu + a \cdot f(x) + b \cdot U() \cdot \sigma + c$, where $f(x)$ is the general shape of the time series without noise, ν the mean value of the series, σ the standard deviation, and $U()$ represents a variate from a uniform random number generator. The parameters a, b and c control the slope (or amplitude), noise and discontinuities. This approach has been successful [18] for generating control time series for clustering or classification algorithms, since it is possible to synthesize time series that have similar shape. Markovian processes may be used to capture dependencies between consecutive elements in sequences. The behavior of such a sequence is summarized in a probability transition table that can be used to process trajectories.

For complex distributions, there are a variety of empirical methods that seek to generate stationary sequences with a given marginal distribution. Some examples [13] are gamma, exponential, Laplace marginals, residual processes, etc. Furthermore, these methods can be classified as background, foreground/background and foreground [14] methods. The Transform-Expand-Sample (TES) methodology is a generalization of these methods and captures autocorrelated variates with Markovian structure.

Adaptive Mimic Functions: A mimic function [20] is a simple device used to mimic a given context — typically textual — in which a hidden message can reside. Given authentic covertext, new covertext is synthesized based on certain semantic rules and bit-sequences from a secret message. Briefly, to enable a mimic function: first build a statistical profile of sample covertext, and then generate a Huffman compression [6] tree. By applying an inversion of the Huffman compression, based on some mapping of the hidden message to bits, new covertext — coarsely resembling the original covertext — is generated. We use the term "coarsely" because this synthetic data generation scheme has a drawback: it exhibits semantic inconsistencies that are easily identifiable to human observers, thus revealing the potential presence of stego.

A possible enhancement to the above procedure is to use a context-free grammar (CFG) with weighed choices on each production rule. These choices are arranged in a decision tree. Each node of the tree will have exactly two branches — one labeled zero (0) and the other labeled one (1). The message to be hidden is transformed into symbols from set $\{0, 1\}$ using some appropriate convention. Then, rules of the grammar are used to produce sentences, where each bit indicates which branch is to be followed in the decision tree. The sentence quality of the grammar depends on the quality of the CFG. It is reported that results can be good, especially for unstructured grammars, including applications in sports news, poetry and spam mail [21]. At the receiving end, the message is decoded by parsing and traversing the decision tree backwards, to obtain the original bits.

The CFG is usually written by hand, a time consuming and detailed procedure. During this process, one has to carefully consider the original data context and synthesize new stego objects in a way that makes them sufficiently rich and well-structured to escape undue attention. We caution that naive Markovian generation schemes are limited in their ability to generate meaningful data paths. That is, empirical transition probabilities lead to covertext data trajectories that are often "far" from the original covertext in a visual, or otherwise measurable sense. This is a failing that can be exploited by detection algorithms.

One way to successfully hide a message inside numerical stochastic data is to proceed as follows. First, we build a statistical profile of the observed information in the selected context (either text- , visual- or numeric-based) using a sufficiently long history. Next, we develop a mapping between numbers and grammar symbols using a discretization constant q. A number x_i which falls in the interval $nq \leq x_i < (n+1)q$ is mapped into the n-th symbol. We use sufficiently long data history to build an order-m Markov chain, with the goal of estimating a probability transition matrix describing transitions between (sets of) symbols. In essence, the production rules of a PCFG encode the probability that a given symbol follows a given substring — which is information that can be extracted from the profile and is summarized in the probability matrix. The final step is to build the production rules and obtain a PCFG which can be used for the mimic functions.

With a good mapping, we can bypass some of the disadvantages of ordinary mimic functions, because it is harder for a human to tell an incorrect data stream from a similar but correct one than it is to tell in the case of language text, simply because of semantics. Without such a need to maintain semantics, grammars need no longer be carefully crafted by hand. The procedure can now be automated, opening up the possibility of generating an unlimited number of grammars. We thus arrive at adaptative mimic functions which can change with the underlying data model.

Limitations: In the context of general and stochastic data steganography, however, mimic functions have certain intrinsic problems. First, the statistical properties of the underlying data are not preserved when a Markovian structure is imposed upon empirical data that may understandably come from a non-Markovian source. Strong local correlations are ignored and replaced by probabilities that average information over the entire data set. Second, the transformation from CFG to a Huffman tree automatically alters the transition probabilities, since the new probabilities are only an approximation. Finally, a given synthesized trajectory — only one among many potential trajectories — is effectively constrained by a message bitstring which cannot be expected to subscribe to the transition probability structure of the data source. That means sufficiently long messages will immediately reveal (statistical) departures from authentic covertext. In terms of steganalysis, every such failure to maintain structure that is "similar" to covertext structure, in some well-defined sense, is another avenue that a detection algorithm might exploit. Thus, to investigate stronger stego methods and perversely, stronger detection algorithms — the main goal of steganalysis — we must look for more stringent ways of information hiding.

2.2 TES-Process Synthesis

The TES (*transform-expand-sample*) [15, 16] process is a generative methodology that enables us to create general, stationary, empirical time series — such as those driven by applications and processes that can harbor stego — with autocorrelations. TES is readily able to capture first and second order properties of series and closely approximate empirical autocorrelations. It has been applied in diverse applications, including compressed video, finance, images/textures and simulation. While TES exploits iid variates for a key constructive component called an *innovation* density, autoregressive modular processes enable us to migrate upwards to dependent innovation sequences, thus generalizing TES processes. For convenience and ease of exposition, we will restrict the discussion to independent innovations. TES and its generalizations, with strong theoretical foundation [7, 8] based on Markov processes and autoregressive schemes, are broadly applicable to arbitrary empirical data. Briefly, given empirical data or *base data* from some application or process, TES-based Monte Carlo enables us to obtain statistical replicas or *clone data* which may then be exploited as application or process *coverdata*. In stochastic settings, there is no known way to differentiate between TES-process data and base data. Given the base data $\{D_n\}_{n=0}^{\infty}$, multiple versions of the base data trajectory can be synthesized with remarkable similarity. In the sequel, we will work with a typical synthesized trajectory $\{X_n\}_{n=0}^{\infty}$. The TES mimicing procedure is implemented as follows:

1 Choose an innovation density f_v, which is used to generate an innovation sequence $\{V_n\}_{n=0}^{\infty}$ by means of a Monte Carlo simulation.

2) Using a TES process and the innovation, generate an autocorrelated but uniform TES background sequence $\{Y_n\}_{n=0}^{\infty}$.

3) Use a stitching transformation $S|\xi$ with chosen parameter ξ, to smooth the background sequence by removing data discontinuities ("jumps").

4) Estimate the discrete cumulative distribution function (CDF) of the base data $\{D_n\}_{n=0}^{\infty}$ and apply a probability integral transform to obtain a foreground sequence $\{X_n\}_{n=0}^{\infty}$.

5) Both mathematically and visually, compare the autocorrelation and structure, respectively, of the forward sequence to the base data. If the error between the two is acceptable, output the forward sequence as an acceptable synthetic time series $\{X_n\}_{n=0}^{\infty}$. If the error is unacceptable, repeat the procedure, modifying V_n and ξ appropriately.

A TES model will meet three requirements, in descending order of rigor:

- The marginal distributions of the base data and synthesized data match (guaranteed by step 4).
- The error between the significant leading autocorrelations is minimal (through innovation density, step 1).
- The trajectory of the synthesized data resembles that of the base data, from an expert's viewpoint (from previous conditions and stitching parameter, step 3).

Since the marginal distribution is guaranteed by the integral transform, TES can then be interpreted as a systematic search for special data sequences in a space parametrized by (V_n, ξ). Such sequences, which must adhere to given autocorrelation and visual resemblance requirements, can be located heuristically, algorithmically [9], or both.

3. REFERENCES

[1] R. J. Anderson and F. A. Petitcolas. On the limits of steganography. *IEEE Journal of Selected Areas in Communications. Special Issue on Copyright and Privacy Communication*, 16(4):474–481, May 1998.

[2] F. P. (Ed.) and S. K. (Ed.). *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, Norwood, MA, 1999.

[3] C. C. et al. A taxonomy of obfuscating transforms. Technical report 148, Department of Computer science, University of Auckland, 1997.

[4] C. C. et al. Manufacturing cheap, resilient and stealthy opaque constructs. In *Proceedings ACM Symposyum on Principles of Programming languages*, pages 184–196, San Diego, California, United States, 1998. ACM Press.

[5] T. Handel and M. Sandford. Data hiding in the OSI network model. In *Proceedings of Information Hiding: First International Workshop*, pages 73–93, Cambridge, U.K., 1996. Springer-Verlag.

[6] D. Huffman. A method for the construction of minimum redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40:1098–1191, 1951.

[7] D. Jagerman and B. Melamed. The transition and autocorrelation structure of TES processes; part I: General theory. *Stochastic Models*, 8(2):193–219, 1992.

[8] D. Jagerman and B. Melamed. The transition and autocorrelation structure of TES processes; part II: Special cases. *Stochastic Models*, 8(3):499–527, 1992.

[9] P. Jelenkovic and B. Melamed. Algorithmic modeling of TES processes. *IEEE Trans. on Automatic Control*, 40(7):1305–1312, 1995.

[10] S. Katzenbeisser and F. P. (eds.). *Information Hiding: Techniques for steganography and digital watermarking*. Artech House, Norwood, MA, 2000.

[11] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 1982.

[12] L. M. Leemis. Input modeling. In *Proceedings of the 26th Winter Simulation Conference*, pages 55–61, San Diego, CA, 1994. Society for Computer Simulation International.

[13] P. Lewis. Some simple models for continuous variate time series. *Water Resources Bulletin*, pages 635–644, 1985.

[14] B. Melamed. TES: A class of methods for generating autocorrelated uniform variates. *ORSA Journal on Computing*, 3:317–329, 1991.

[15] B. Melamed. An overview of TES processes and modeling methodology. In L. Donatiello and R. Nelson, editors, *Performance Evaluation of Computer and Communication Systems*, pages 359–393. Springer-Verlag Lecture Notes in Computer Science, 1993.

[16] B. Melamed. The empirical TES methodology: Modeling empirical time series. *J. of Applied Mathematics and Stochastic Analysis*, 10(4):333–353, 1997.

[17] S. Murdoch and S. Lewis. Embedding covert chanels into TCP/IP. Technical report. Technical report, Security Group of the University of Cambridge, 2005.

[18] A. Nanopoulos, R. Alcock, and Y. Manolopoulos. Feature-based classification of time-series data. *Information processing and technology*, pages 49–61, 2001.

[19] R. C. Summers. *Secure Computing Threats and Safeguards*. McGraw-Hill, Boston, MA, 1997.

[20] P. Wayner. Mimic functions. *Cryptologia*, 16(3):193–214, 1992.

[21] P. Wayner. *Dissapearing Cryptography. Information Hiding: Steganography & Watermarking*. Morgan Kaufmann, San Francisco, CA, second edition, 2002.

[22] A. Westfeld and G. Wolf. Steganography in a videoconferencing system. In *Proceedings of 2nd Intl. Workshop in Information Hiding. Lecture Notes in Computer Science*, volume 1525, pages 32–47, Portland, Oregon, 1998. Springer-Verlag.