Combining Steganography and Zero-Knowledge Proofs to Embed and Prove a Digital Signature in an Image

Matthew Toso Sun B. Chung
Quantitative Methods and Computer Science
University of St. Thomas
Saint Paul, MN 55105
mdtoso@stthomas.edu, sbchung@stthomas.edu

Abstract

Steganography is a field of research involving hiding information in a seemingly innocuous cover message. This concept is applicable to digital images, audio, and video which have pixels and streams in which secret data can be embedded in ways that make no perceivable difference, even when using statistical analysis. An implementation of steganography is presented for embedding a digital signature in an image to prove its ownership.

Zero knowledge proofs (ZKP) are techniques for proving identity or ownership. Their fascinating property is that possession of some information can be proved without revealing that information. This is done in an interactive process whereby a prover convinces a verifier of a certain assertion.

In this paper, we combine the concepts of steganography and ZKP in order to create a digital signature scheme in which no one gains enough information to falsely claim ownership.

1. Steganography

1.1 History and Definition

In the fifth century B.C. the Spartans fell to the invading Persian army in the famous battle of Thermopylae. The exiled king Demaratus sought to warn the Spartans who remained at home that the Persian army was ready to march against them. In order to send a message unknown to the Persians he took a wax tablet in common use for writing at the time, removed the wax, wrote his message in the wood underneath, and again covered the tablet with wax. The message passed the Persian guards unhindered and was discovered by King Leonidas's wife once it reached Greece [1].

If true, this story as given by the Greek historian Herodotus is one of the earliest known uses of steganography. Steganography is the practice of hiding a message so that it can pass undetected. Generally this is accomplished by placing the message within a larger innocuous message. The actual message is called the "payload" and the larger message hiding it is called the "cover message". It is a concept related to cryptology but different from simple encryption which is concerned with making a message undecipherable. A well-encrypted message is not readable but anyone obtaining the message will realize that sensitive information is being passed. On the other hand, with steganography, it is likely no one will even realize the valuable information is being exchanged.

There are many examples in history where steganography was used because of this vital property. It was vital for 1st Lieutenant Jonelis during his time in a Japanese P.O.W. camp during World War II. He addressed a postcard to Mr. F. B. Iers in Los Angeles with the following message:

Dear Iers:
After surrender, health improved
Fifty percent. Better food, etc.
Americans lost confidence
In Philippines. Am comfortable
In Nippon. Mother: invest
30%, salary, in business. Love
Frank G. Jonelis

Figure 1: A postcard sent from a Japanese POW camp to the FBI.

The message passed the Japanese censor without difficulty and eventually reached the FBI's office in Los Angeles. It appears to be a completely harmless description of the Lieutenant's health and some investment advice. Read only the first two words of each line, however, and it is discovered to in fact be intelligence about American war losses, "After surrender fifty percent Americans lost in Philippines in Nippon 30%" [2].

1.2 Applications in the Digital Age

From a layer of wax to selective words and invisible ink, steganography has progressed much since the time of the ancient Greeks. The most exciting developments, however, have occurred only in the past several decades through the pervasiveness of the computer. In the analog world, there are many places in which a message can be hidden; certain letters might be written more boldly than others or the type set slightly lower for example. In the digital world, there is only one place to hide data: a bitstream of zeros and ones. This is not to say that the applications of steganography to digital data are limited; in fact the opposite is true. While message hiding was generally limited to some form of text, today a message can be hidden in anything that can be saved digitally. Any digital file, whether it be text, a photograph, audio, or video, is a potential cover message. The boundaries of what the payload can be are extended in the same way. Instead of simple text, one could embed, for example, a digital signature into a file.

One important consideration when choosing an appropriate cover message for a file is the resulting "encoding density". Encoding density is a number between 0 and 1 calculated by dividing the number of bits in the payload by the number of bits in the cover message. An encoding density near 1 results in the smallest possible resulting message, but usually the smallest possible encoding density is most desirable. A low encoding density means fewer changes to the cover message are needed. This in turn makes the hidden file less obvious both to the human eye/ear and to statistical analysis which, when applied to stego-files, is called steganalysis.

Digital images provide ideal cover messages. For one, they are large. An image will commonly use 8 bits to represent each of the three color components red, green, and blue per pixel resulting in a total of 24 bits per pixel. This, multiplied by the millions of pixels taken by modern digital cameras, results in a file size of many megabytes. Another reason digital images work so well as cover messages is the amount of noise present. Text, executables, and many other types of files are full of patterns that, if tampered with even slightly can be very noticeable. On the other hand, changes to images are difficult to notice among noise created by film grain, unclean lenses and lossy compression among other things.

For the sake of illustration, consider the uncompressed 200 by 200 pixel image of a tree shown in figure 2. Hiding data in the 2 lowest order bits of each color component results in an encoding density of .25 and 30 kilobytes of possible hidden data. Through changes unlikely to be visible to the human eye, this small image of a tree has the ability to hide a long text message. In this case, it is in fact hiding a second image of the same dimensions! Setting the 6 high order bits to 0 will result in an image that looks almost black, but making this resulting image 85 times brighter reveals the hidden photo of a cat [3].

Figure 2: An image of a tree and the photograph extracted from it.
Used under the terms of the GNU Free Documentation License.

In practice, however, an encoding density smaller than .04 will generally be used. This allows for use of less than one bit per pixel for encoding and greatly increases the difficulty of detecting the payload while still allowing for a reasonable payload size.

2. Zero Knowledge Proofs

Encryption frequently plays an important role in digital steganography. Some implementations will encrypt payloads before embedding them within a file for two reasons. The first is that encrypted messages are harder to extract. If an unintended recipient somehow manages to extract the payload from the cover message, he or she must still contend with the imposing task of decryption. The second reason is to make the payload seem more like randomly distributed noise. In most modern encryption schemes, a cipher text has a seemingly random distribution of characters. Patterns in payloads such as ASCII text are therefore eliminated by encryption, making the data much more difficult to detect.

In this section we describe an alternative scheme to embed a digital signature in an image that does not require encryption but has the powerful property of not revealing crucial information about the ownership of that image.

2.1 Description and Illustration

A zero knowledge proof (ZKP) allows one to convince someone of the validity of a statement without revealing anything else, even why the statement is true! To one who has never heard of a zero knowledge proof, this concept can seem unintuitive but mathematics makes it possible.

There is, in fact, an often-used story [4] to help one visualize how a zero knowledge proof works. For our discussion we are concerned with two parties, a prover and a verifier. The prover tries to convince the verifier of some assertion. In this story we will call the

prover Peggy and the verifier Victor. It takes place at the entrance to a cave with a circular passageway. One could pick a direction, walk into the cave, and end up back at the entrance if it weren't for the existence of a closed magic door on the far side. Peggy says she knows a magic word that will open this door and is willing to sell this secret to Victor. Victor won't pay until he knows Peggy is telling the truth and Peggy won't tell Victor her secret until she gets paid.

They seem to be at an impasse but a strategy is devised in which Peggy can convince Victor that she knows the magic word. Victor looks away as Peggy chooses one of the paths into the cave. He then tells Peggy which side he wants her to come out on. If she really knows the magic word, she can always come out on the correct path. If Peggy does not know the magic word, she could play the game anyway hoping to swindle Victor. She would not know which path Victor will pick beforehand, however, so there is a probability of 1/2 that she will be revealed as a cheater. Since after one round there is still a reasonable probability that Peggy is cheating, they repeat the process enough times to convince Victor that Peggy truly knows the magic word. After several rounds, the chances that Peggy is cheating become vanishingly small. If ten rounds of the cave game were played, the chances Peggy is cheating are 1/1,024 or less than 0.1% (and less than 1/1,000,000 after twenty rounds).

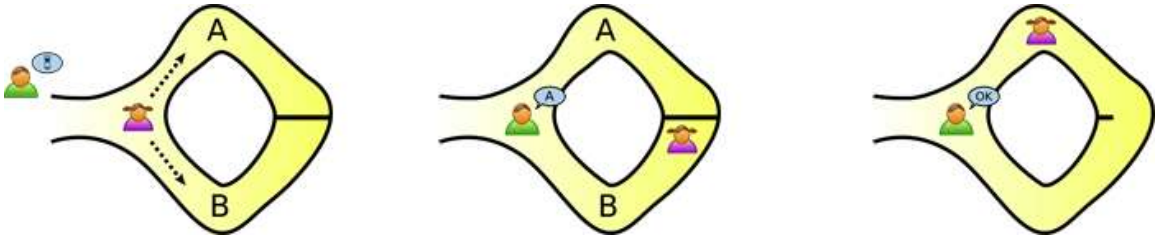


Figure 3: Peggy and Victor complete a zero knowledge proof.

2.2 Definition

Now that the basic functioning of a ZKP is understood, it is time to define them in mathematical terms. It can be seen from the cave example that the proof works differently from a standard, static proof. The verifier does not stand idly by while the prover works through a set of predetermined computations. Instead, the proof is an *interactive* process in which the prover must take certain courses of action depending on the verifier's input. This is called an interactive proof.

The functioning of an interactive proof is often described as an interaction between two Turing machines. The machines share a common input tape and can send messages to one another through a communication tape. The machines can see these two tapes as well as their own. This means that a machine can communicate with the second machine but cannot monitor its current state, program, or any other of its internal workings [5].

A precise definition of an interactive proof is given by Goldwasser *et al.* [6] as follows:

> An interactive proof system for a set S is a two-party game, between a *verifier* executing a probabilistic polynomial-time strategy (denoted V) and a *prover* which executes a computationally unbounded strategy (denoted P), satisfying
> - Completeness: For every $x \in S$ the verifier V always accepts after interacting with the prover P on common input x.
> - Soundness: For some polynomial p, it holds that for every $x \notin S$ and every potential strategy P*, the verifier V rejects with probability at least $1/p(|x|)$, after interacting with P* on common input x.
>
> The class of problems having interactive proof systems is denoted IP.

Loosely speaking, completeness means that an honest verifier will always be convinced of true statements by an honest prover. Soundness means that, with a high probability, the verifier will always reject a cheating prover. In a coin-flip, for example, a cheating prover will be rejected with probability 1/2 in each round. In most strategies, the chance of rejecting a cheating prover is even higher. All Zero Knowledge Proofs of interest are interactive and therefore must satisfy these conditions of completeness and soundness.

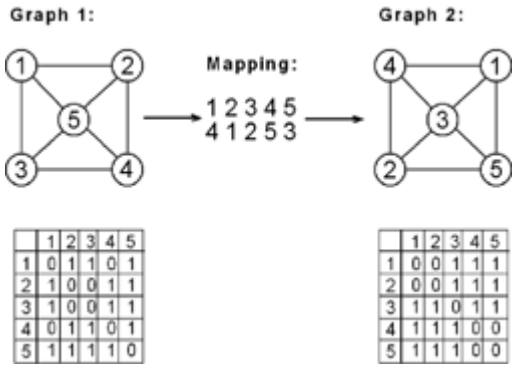
There is a third condition that a ZKP must satisfy called, appropriately enough, zero-knowledge. It is a property of the interactive process defined by considering what can be computed by an *arbitrary* feasible verifier interacting with the prover on a common input taken from the set of valid assertions. This "is compared against what can be computed by an *arbitrary* feasible algorithm that is only given the input itself" [7]. An interactive proof is zero-knowledge if what can be computed in both cases is computationally indistinguishable. The verifier can try any strategy it likes to learn the prover's secret but, after any possible interaction, it learns nothing more than it could have learned from the common input taken by itself. The term zero-knowledge can be somewhat deceptive because a ZKP reveals one *bit* of information to the verifier: that the prover's assertion is valid.

2.3 An Example of a Zero Knowledge Proof

In order to implement a ZKP in software, the descriptions of physical people and a cave need to be translated to abstract mathematical structures. Instead of Peggy and Victor we now will be concerned with the interaction of two computers. The word computer is meant as any device that can run algorithms in polynomial time including personal computers, ATMs, smart cards, and embedded devices among others. Let their common input be two undirected, isomorphic graphs. The secret is no longer how to pass from one side of the cave to the other but how to transform one graph into the other. This is called a zero-knowledge proof for graph isomorphism [5].

It is in fact easy to transform one graph into a given isomorphism if the correct mapping is known. To do this, a program would simply iterate through all vertices, replacing each with the equivalent vertex given in the mapping. Without this mapping, however, it is

difficult to discover how to transform one graph into another or to determine whether the transformation is possible at all. Using small graphs, it would be reasonable to find the isomorphism in a brute force manner but, as the number of vertices increases, the problem quickly becomes difficult.



	1	2	3	4	5
1	0	1	1	0	1
2	1	0	0	1	1
3	1	0	0	1	1
4	0	1	1	0	1
5	1	1	1	1	0

	1	2	3	4	5
1	0	0	1	1	1
2	0	0	1	1	1
3	1	1	0	1	1
4	1	1	1	0	0
5	1	1	1	0	0

Figure 4: Two small isomorphic graphs, their mapping, and their matrix representation.

Let the two isomorphic graphs be labeled G1 and G2. In this ZKP, the prover's goal is to convince the verifier that she knows the isomorphism between G1 and G2 without ever revealing the isomorphism. The prover starts by generating a third graph H as a random isomorphic copy of G1 and sending H to the verifier. The verifier then tosses a coin and randomly asks the prover for one of two things; the isomorphism between graphs G1 and H or the isomorphism between graphs G2 and H. The prover can easily provide the isomorphism between G1 and H since it generated H from G1. It can also quickly calculate and send the isomorphism between G2 and H by taking the composition of the G1:H and G1:G2 isomorphisms. The verifier checks to ensure the isomorphism it requested is valid and the process is repeated n times, where n can be chosen to satisfy any confidence level required.

It is easy to verify that the above proof is indeed zero-knowledge. In each round the verifier can receive only one of two things: an isomorphism between a random graph and G1 or an isomorphism between a random graph and G2. Receiving only one of these is useless in helping the verifier solve the secret isomorphism between G1 and G2. Its only application is to further convince the verifier that the prover knows the secret. By verifying a G1:H isomorphism, the verifier is further convinced that the prover is not creating bogus H's. By verifying a G2:H isomorphism, the verifier is further convinced that the prover knows the G1:G2 isomorphism.

This is an example of a non-deterministic polynomial (NP) problem. Given a problem and its solution, it is easy to verify that the solution is valid. The graph isomorphism problem is one of many examples of NP problems for which the solution is difficult to find. It has been shown [7] that every problem in NP has a zero-knowledge interactive proof.

3. A New Digital Signature Scheme

The basic concepts of steganography and zero knowledge proofs have been described as well as how their use applies to computers. Now the two ideas will be combined to create a powerful digital signature scheme.

3.1 How Digital Signatures Prove Ownership

A digital signature in a digital file is analogous to a handwritten signature on paper. Like a handwritten signature, it uniquely identifies an individual. Good signatures allow no one to impersonate another. A file given a digital signature allows the owner and only the owner of that file to claim ownership.

The common input to a zero knowledge proof makes a good digital signature because this property is guaranteed. Only the true owner will be able to claim ownership of a file including the common input because she is the only one who can prove the statement given. Ownership can be proved as many times as needed to any number of users and, because it is zero-knowledge, no one gains the ability to later impersonate the owner.

The graph isomorphism problem described earlier can easily be represented in a way suitable for a signature. G1 and G2 can be stored as an "adjacency matrix" [8]. The matrix is filled with 1s and 0s with the row and column numbers indicating vertex numbers. A 1 indicates that an edge exists between the two corresponding vertices while a 0 indicates that no edge exists. Note that the cells along the diagonal will always be 0 because we are not considering vertices that connect to themselves. Also, there is some redundant data which can be thrown out. Data are mirrored across the diagonal because G1 and G2 are undirected.

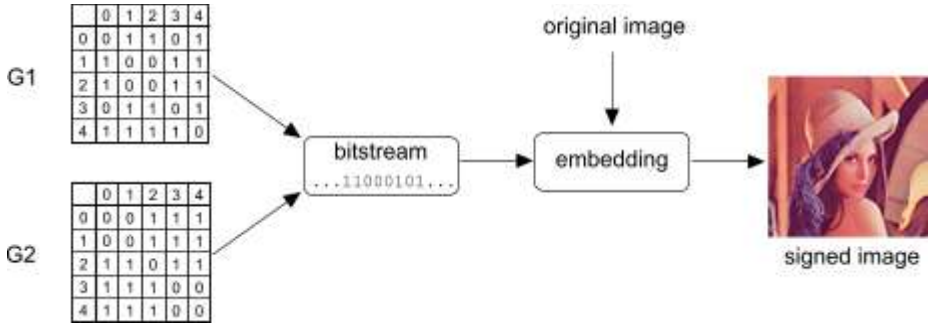


Figure 5: A process for signing a digital photograph.

There remains the problem of where this digital signature should be placed. For an image, steganography provides an ideal answer: hide the digital signature within the pixels of the image itself. Figure 5 shows how the graph isomorphism problem and steganography can be combined to digitally sign an image.

4. Conclusion and Future Work

Steganography, the practice of hiding a message within a message, and zero-knowledge proofs, a way to prove an assertion without revealing why it is true, combine to form a powerful approach to digitally signing an image. In particular, the graph isomorphism problem can have its common input graphs represented as adjacency matrices to be embedded into an image thus proving ownership. As all problems in NP have ZKP's, however, there are a great deal of possible variations on this approach.

Future work will involve analyzing these other approaches to determine which one is best suited for a digital signature scheme. It is very likely that the graph isomorphism problem does not provide the highest possible level of security. When the vertex degree is bounded by a constant, there is a polynomial time algorithm in which the isomorphism can be found. While still being a difficult problem for a large number of vertices and edges, problems that can be solved in polynomial time are not among the most difficult in computing.

NP-complete problems are good candidates as they are the most difficult problems in NP to solve. Two examples are the Hamiltonian cycle and graph three-coloring problems, both of which have known ZKP's. The choice of a proof is complicated by the fact that even NP-complete problems such as these have been shown to be easily computable for structures that are not carefully constructed [9,10].

There is also further research to be made into the subject of steganography. Under most digital image steganography schemes, tampering with the image will damage or destroy the embedded text. In the case of a JPEG image, for example, something as simple as opening then saving the file could erase any embedded date because the image will be recompressed. It would be interesting to research methods to make a stego-image resistant to tampering as well as to restore damaged data.

References

[1] F. Godolphin, *The Complete and Unabridged Historical Works of Herodotus*, New York: Random House, 1942.

[2] R. Kippenhahn, *Code Breaking: A History and Exploration*, Woodstock: Overlook Press, 1999.

[3] "Steganography," [Online document], 2006 Mar 6, [cited 2006 Mar 7], Available HTTP: http://en.wikipedia.org/wiki/Steganography

[4] "Zero Knowledge Proof," [Online document], 2006 February 25, [cited 2006 Mar 7], Available HTTP: http://en.wikipedia.org/wiki/Zero_knowledge_proof

[5] O. Goldreich, S. Micali, A. Wigderson, "Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design," *Proceedings of*

the 27th Annual Symposium on Foundations of Computer Science, pp. 174 – 182, 1986.

[6] S. Goldwasser, S. Micali and C. Rackoff, "The Knowledge Complexity of Interactive Proof Systems," *SIAM Journal on Computing*, vol. 18, pp. 186-208, 1989.

[7] O. Goldreich, "Zero-Knowledge Twenty Years After its Invention," *Weizmann Institute of Science*, 2002.

[8] H. Peelle, "Graph Coloring in J: An Introduction," *Proceedings of the APL 2001 Conference*, 2001.

[9] A. Broder, A. Frieze, E. Shamir, "Finding Hidden Hamiltonian Cycles," *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pp.181 – 189, 1991.

[10] B. Hayes, "On the Threshold," *American Scientist*, 2003.