OutGuess - Practical Steganography

Niels Provos

provos@citi.umich.edu

Center for Information Technology Integration

University of Michigan

Ann Arbor

 Copyright Niels Provos 1999

Overview

1. Introduction
2. Prerequisites
3. Construction
4. Minimizing Modifications
5. Analysis and Future Work
6. Conclusion

 Copyright Niels Provos 1999

Introduction

Steganography hides the fact that communication is happening, *e.g.* holiday picture with hidden message in least significant bits.

We say:

> a **hidden message** is embedded into the redundant data of the **cover-medium** to create the **stego-medium**.

Introduction

- Steganographic systems
  - *classical*: used to rely on keeping encoding system secret
  - *modern*: only detectable if secret information known
- Invasive nature of systems might leave detectable traces. Secrecy of the information is not degraded, but its hidden nature is revealed, defeating purpose of steganography.
- **How to do it better**?

 Copyright Niels Provos 1999

Introduction

Minimize required modifications:

- for a specific hidden message, find image that matches the data the best
- other tricks:
  - probabilistic embedding
  - error-correcting codes
  - plausible deniability, ...

$\rightarrow$ Need flexible framework to play with: OutGuess

Prerequisites

- Steganography remains undetected only if original image is unknown. Otherwise comparison to original reveals changes.
- **Easy** in practice.
- Create your own pictures with digital cameras, nobody will suspect anything when you send holiday pictures to friends.

 Copyright Niels Provos 1999

Construction

- Split embedding into two separate steps:
  - extraction of redundant data; can be modified without destroying cover medium
  - selection of bits in which hidden data will be placed
- Extraction process passes information for each bit to selector:
  - how *detectable* are changes to this bit
  - is this bit *locked*, *i.e.* can it be changed?

Construction

Extraction Process

Redundant data is extracted, modified and then written back.

The extraction process

- depends heavily on data format, might be non-deterministic
- for JPEG use the LSB in the discrete cosine transform coefficients

 Copyright Niels Provos 1999

Construction

Locked Bits

A Bit

- that is locked may not be modified
- becomes locked when it has been used to carry information, or modification destroys content of cover medium

 Copyright Niels Provos 1999

Construction

Detectability

- Detectability is used as bias in selection process
- selector tries to minimize modified bits and detectability of changes
- different embeddings have different accumulated bias, the one with lowest detectability wins

 Copyright Niels Provos 1999

Construction

Selection Process

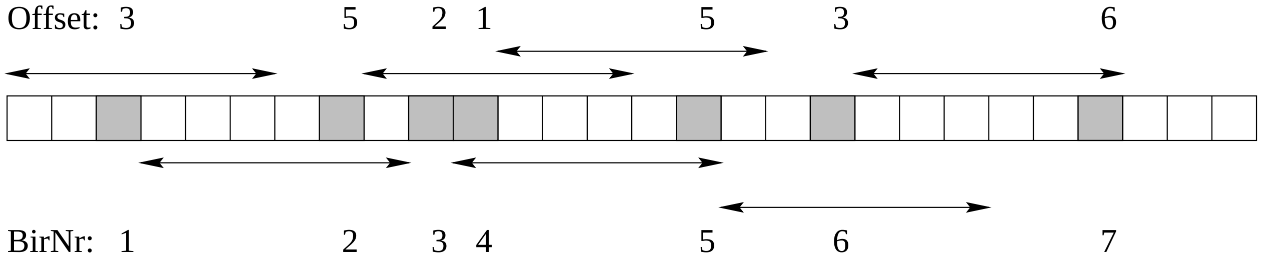
Alleged RC4 stream is

- used to encrypt hidden message
- used to create PRNG (pseudo random number generator)
- initialized with user-chosen secret key

Selection with iterator function using the PRNG.

 Copyright Niels Provos 1999

Construction



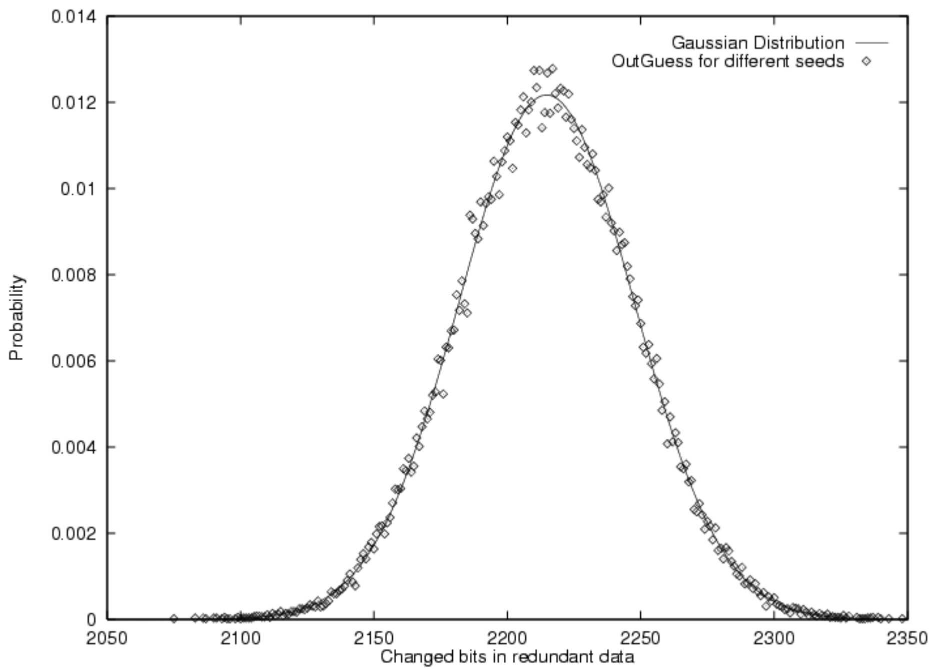
- Iterator is initialized
  - with a bitmap (the extracted redundant data)
  - and the PRNG.
- Each call returns the next selected bit. All returned bits are evenly spread over bitmap.

Construction

- Iterator can be reseeded to adapt its behaviour, resulting in different embeddings for each different seed.
- Next bit is selected within interval. The interval depends on the size of the remaining data.
- Hidden message is prepended with header containing
  - a 16-bit seed and
  - the length of the hidden message

 Copyright Niels Provos 1999

Minimizing Modifications



Seed creates a family of PRNGs.

- Number of changed bits is Gauss-distributed.
- Mean is close to 50% of bits in hidden message.

 Copyright Niels Provos 1999

Minimizing Modifications

- Iterator can be reseeded with a freely chosen seed.
- Seed at lower end of Gaussian distribution reduces number of bits that need to be modified.
- Unfortunately, we observe $\frac{\sigma^2}{\mu} \approx const.$
- Varying the seed allows us to
  - skip locked bits
  - hide multiple data sets

 Copyright Niels Provos 1999

Minimizing Modifications

- Modifications of locked or detectable bits can be avoided by introducing errors into hidden data
- requires error-correcting codes: $[n, k, d]$
- t errors can be corrected, with $d = 2t + 1$
- **but** error encoding increases size of data

 Copyright Niels Provos 1999

Minimizing Modifications

- number of modified bits remains the same, if
  - $\frac{n}{2} - t = \frac{k}{2}$ or
  - $d = n - k + 1$ holds.
- those are the maximum separable distance codes (MDS)
- repetition code $[n, 1, n]$ is the only binary MDS code.
- choose to not modify t bits from each code block

 Copyright Niels Provos 1999

Minimizing Modifications

- a more knowledgeable observer might detect modifications and conclude use of steganography
- multiple hidden messages in stego-medium provide **plausible deniability** if one of them is inconspicuous
- framework supports multiple embeddings with locking bits and new iterations for each message
- probabilistic embedding and error correction allow us to skip conflicts on locked bits

 Copyright Niels Provos 1999

Analysis and Future Work

- analyse embedding for JPEG
  - modifications of redundant data could lead to impossible DCT coefficients
  - how does the detectability heuristic help to avoid detection?
  - can heuristic be improved by analysis of possible DCT coefficients?
- image selection to find image with best representation of hidden message

 Copyright Niels Provos 1999

Conclusion

- probabilistic embedding and error correction with MDS codes are new contributions to steganography
- OutGuess is available at
  `http://www.outguess.org/`
- Acknowledgments
  - Andrew Carra for early discussions on minimizing modifications
  - Sean Coffey for help on error-correcting codes

 Copyright Niels Provos 1999