Proceedings of the 10th International Conference on Applied Informatics
Eger, Hungary, January 30–February 1, 2017. pp. 157–162
doi: 10.14794/ICAI.10.2017.157

Source Code-based Steganography

Dávid Kis, Norbert Pataki

Dept. of Programming Languages and Compilers
Fac. of Informatics, Eötvös Loránd University, Budapest
`{kidraai,patakino}@inf.elte.hu`

Abstract

Steganography is an approach that aims hiding secret messages into a visible, readable carrier or cover. The secret message is typically not encrypted but it cannot be read by everyone because most of the people do not know that this stegotext contains secret messages and they do not know how to find it. The most common carrier types are images, audio and video files.

In this paper we present an approach that aims at steganography at the level of source code, written in different programming languages. Nowadays many open source development repositories are available, huge code legacies can be taken advantage of and internet portals gain popularity and offer opportunity to share hidden messages. However, these codes are sequence of characters that is not a typical medium of steganotext. We present different techniques that can be used for source code-based steganograpy. This means that behaviour of the code remains the same but it includes a hidden message. High-level programming languages support different constructs with the same result and we take advantage of these constructs. However, we cover general and specific techniques as well.

Keywords: Steganograpy, source code

MSC: 68N30 Mathematical aspects of software engineering

1. Introduction

Steganography is a concept which aims at hiding secret into visible, readable message. The containing message is called carrier or cover. The secret message is typically not encrypted but it cannot be read by everyone because most of the people do not know that this stegotext contains secret messages and they do not know how to find it. Steganography uses many different techniques for hiding messages. This approach has been used for thousands of years, the ancient Greeks have invented the very first techniques.

In the digital era the steganography has been changed. Images have become the most popular target of holding secret messages. However, audio and videos are also can be used.

Nowadays many open source development repositories are available, huge code legacies can be taken advantage of and internet portals gain popularity and offer opportunity to share hidden messages. However, these codes are sequence of characters that is not a typical medium of steganotext. We present different techniques that can be used for source code-based steganograpy. This means that behaviour of the code remains the same but it includes a hidden message. High-level programming languages support different constructs with the same result and we take advantage of these constructs. However, we cover general and specific techniques as well.

2. Steganography

Steganography is a set of techniques that hides message into a visible, readable carrier or cover. The word *steganography* combines the Greek words steganos meaning "covered, concealed, or protected" and graphein meaning "writing". Steganography techniques have been used for thousands of years and many techniques have been developed. The ancient Greeks invented the very first techniques, for example a slave's hair is cut, the message is tattoed to his head, and when the hair is grown the slave was the messenger. The steganograpy aims hiding secret messages that cannot be realized when it is found.

Nowadays, images are typically used as covers. There are many different techniques to hide secret message into digital images [8]. Spatial domain and frequency domain approaches are well-known ones, but masking and filtering is also common steganography method [7]. The spatial domain approach uses encoding in Least Significant Bits [3]. This approach utilitizes that the human eyes cannot detect the variations in the LSB (see figure 1). Different LSB methods can be mentioned:

- LSB Replacement – the cover image's LSBs are substitued with a bit of hidden message's bit
- LSB Matching – covers some occasional substraction and addition operations
- Matrix embedding – both images are changed due to randomly changed LSBs can be detected.

Frequency domain steganography uses JPEG format because the size of files is relatively small. JPEG compression involves various steps like conversion of RGB to brightness, chrominance and color. During this conversion mathematical methods (e.g. Discrete Cosine Transformation) are applied. This approach affects the coefficients. Different techniques are available: Discrete Wavelet transformation, Discrete cosine transformation, Discrete Fourier transformation [7].

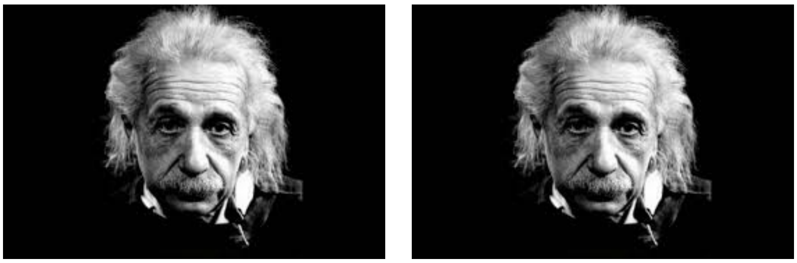
Figure 1: Image without and with secret message

Masking and filtering is a technique for gray-scale images. This method embeds the secret message particularly in more significant areas rather than hiding it only into the noisy section [4].

However, the programs' source code is not a typical cover. Nowadays huge online repositories are available, open source software development is a common solution and large codebases are around us where no one imagines hidden messages. Internet portals return an ever-changing HTML, CSS and JavaScript source code to the browser. The source code also can contain special hidden messages. In this case, the source must be changed, but the semantics of code must be untouched. Executable code as cover has been analysed [1]. Office Open XML format can be considered as a cover format [2]. Another astounding cover type is 3D geometries [10].

3. Steganography in Source Code

Common carrier types are images, audio and video files because these formats are durable to noise, secret messages can be implanted into these noise. However, while one bit change in an image is a basically unnoticeable shade change in a pixel, in text files one bit change is a very noticeable character change. That is why text based steganography is not as common as the other carrier types.

In this paper we propose methods for source code-based steganography. Our aim is hiding messages into source code. In this case, we can hide bits, bytes by using specific high-level constructs from the same possibilities. One can hide more complex messages via special comments or identifiers. We distinguish our methods: general and specific approaches. The techniques of the general approaches can be used with a bunch of programming languages, for instance, these can be used with the most of the object-oriented languages or even in more general way. The specific approaches take advantage of constructs of given programming language (e.g. C++).

3.1. General approach

One of the most obvious general methods is to hide data within comments, because most compilers ignore it. This method has a good encoding rate, but if one take a look at the source code, the message is easily detectable. A less detectable method is to associate meaning to the whitespaces and its layout. One can alternate between spaces and tabs which can represent zero and one bits. It would be more cautious to just replace already present whitespace – generally a tab is two or four or eight spaces long depending on the environment –, but one can place arbitrary length messages on line ends. This trick can also be used with alternating between upper-case and lower-case characters in case-insensitive languages.

Generally a good method is to inject meaning into how the source deviates from a given code convention or standard. Most of the conventions specify which case to use with different identifiers, eg. CamelCase, mixedCase, snake_case to use with different identifiers as in variable, function or constant names. Usually the code author is consistently following the same style, but subtle changes, like some method's name is differently cased is easily overlooked.

3.2. Specific approach

Normally an average user rarely takes a look at the currently viewed web-page's source code, that makes it an ideal hiding medium. Opposed to compiled languages, the source of a page is downloaded to anyone, who visits the page. Assuming there is a time sensitive message that needs to be send secretly, one can easily conceive a protocol that contains a hidden message and compliant with html standards so the browsers renders the page as before tampering [12]. One protocol is to utilize that as opposed to xhtml, in html standards the tag and attribute names are case insensitives. The secret message is encoded in utf-8 then the bits determine the cases of the alphabetic characters. To achieve better data integrity at the expense of size and obscurity, the message length and its checksum can be stored at the start with fixed length [11]. Alternatively in analog to the string handling in C, the end of the message can be marked with a previously agreed upon delimiter.

Let us consider the following example:

```
<!dOCtypE hTMl>
<hTmL lANG='en'>
  <heAd>
    <mETa HTTp-eQUiv="Content-Type"
          coNtENT="text/html;charset=utf-8"/>
    <tiTle>
      Doesn't look like anything to me
    </TItlE>
  </head>
</html>
```

Another approach can be mentioned. HTML entites can be used for describing reserved characters in HTML. Entities have numeric and named format to describe the same character, for instance the following entities define the same (non-breaking space and ampersand):

```
&nbsp; &#160;
&amp; &#38;
```

This convention can be used to inject additional bits. If one uses the numeric format it means zero, otherwise one.

3.3. Implementation

We implemented a proof of concept for the html hide in case-insensitiveness technique in python. We subclassed the standard HTMLParser class which provides some useful virtual methods such as `handle_starttag(tag, attrs)` which is called when it parsed a starting tag. Unfortunately the starting tag and its attributes are passed as the parameter in lowercase regardless of the source file. The workaround for that is to read from the source directly using `getpos()` which returns with the line number and offset of the currently parsed tag. For encoding we overwrite the parsed tags and attributes, with the correct casing, for decoding we interpret it as a bitstring.

4. Conclusion

Steganography is set of techniques that include a secret, but visible message into a cover. Steganography has been used for thousands of years. Nowadays, digital realm has realized the advantages of these approaches. A typical cover type of modern steganography is digital images.

In this paper we analysed how the source code can be utilized for steganography. We have developed a proof-of-concept implementation for this reason. This tool deals with HTML source code. On the other hand, we have distinguished general and specific approaches. General approaches are not related to only one language, but can be used in many high-level languages.

References

[1] Anckaert, B., De Sutter, B. Chanet, D., De Bosschere, K.: *Steganography for Executables and Code Transformation Signatures*, Information Security and Cryptology – ICISC 2004, Lecture Notes in Computer Science **3506** (2005), pp. 425–439.

[2] Castiglione A., D'Alessio B., De Santis A., Palmieri F.: *New Steganographic Techniques for the OOXML File Format*, Lecture Notes in Computer Science **6908**, pp. 344–358 (2005).

[3] Cheddad, A., Condell, J., Curran, K., McKevitt, P.: *Digital image steganography: Survey and analysis of current methods*, Signal Processing **90(3)** (2010), pp. 727–752.

[4] Dagadita, M. A., Slusanchi, E. I., Dobre, R.: Data Hiding Using Steganography, IEEE 12th International Symposium in Parallel and Distributed Computing (2013), pp. 159–166.

[5] Huang, H. J., Zhong, H. J., Sun, X. M.: *An Algorithm of Webpage Information Hiding Based on Attributes Permutation*, in Proc. of The Fourth International Conference on intelligent Information Hiding and Multimedia Signal Processing (2008), pp. 257–260.

[6] Johnson, N. F., Jajodia, S.: *Exploring steganography: Seeing the unseen*, Computer **31(2)** (1998), pp. 26–34.

[7] Kaur, A., Kaur, R., Kumar, N.: *A Review on Image Steganography Techniques*, International Journal of Computer Applications **123(4)** (2015), pp. 20–24.

[8] Provos, N., Honeyman, P.: *Hide and Seek: an Introduction to Steganography*, IEEE Security & Privacy **99(3)** (2003), pp. 32–44.

[9] Sun, G.: *An Algorithm of Webpage Information Hiding Based on Class Selectors*, in Proc. of Third International Symposium on Intelligent Information Technology and Security Informatics (IITSI) 2010, pp. 691–694.

[10] Wu, H.-T., Dugelay, J.-L.: *Steganography in 3D Geometries and Images by Adjacent Bin Mapping*, EURASIP Journal on Information Security (2009), Article ID 317165.

[11] Zhang X., Zhao G., Niu P.: *A novel approach of secret hiding in webpage by Bit Grouping Technology*, Journal of Software **7(11)** (2012), pp. 2614–2621.

[12] W3C HTML5: `https://www.w3.org/TR/2014/REC-html5-20141028/`