**International Journal of Informatics and Communication Technology (IJ-ICT)**
Vol. 10, No. 3, December 2021, pp. 188∼197
ISSN: 2252-8776, DOI: 10.11591/ijict.v10i3.pp188-197 

# Forensic steganalysis for identification of steganography software tools using multiple format image

**S. T. Veena[1], S. Arivazhagan[2]**
[1]Department of Computer Science and Engineering, Mepco Schlenk Engineering College, Tamilnadu, India
[2]Department of Electronics and Communication Engineering, Mepco Schlenk Engineering College, Tamilnadu, India

**Article Info**

***Article history:***

Received May 15, 2021
Revised Sep 20, 2021
Accepted Oct 11, 2021

***Keywords:***

Clustering
Image metadata
Signature artefact
Steganographic software tool
Structural steganalysis
Targeted steganalysis
Universal image steganalysis

**ABSTRACT**

Today many steganographic software tools are freely available on the Internet, which helps even callow users to have covert communication through digital images. Targeted structural image steganalysers identify only a particular steganographic software tool by tracing the unique fingerprint left in the stego images by the steganographic process. Image steganalysis proves to be a tough challenging task if the process is blind and universal, the secret payload is very less and the cover image is in lossless compression format. A payload independent universal steganalyser which identifies the steganographic software tools by exploiting the traces of artefacts left in the image and in its metadata for five different image formats is proposed. First, the artefacts in image metadata are identified and clustered to form distinct groups by extended K-means clustering. The group that is identical to the cover is further processed by extracting the artefacts in the image data. This is done by developing a signature of the steganographic software tool from its stego images. They are then matched for steganographic software tool identification. Thus, the steganalyser successfully identifies the stego images in five different image formats, out of which four are lossless, even for a payload of 1 byte. Its performance is also compared with the existing steganalyser software tool.



***Corresponding Author:***

S. T. Veena
Department of Computer Science and Engineering, Mepco Schlenk Engineering College
Sivakasi, Tamilnadu-626005, India
Email: veena_st@mepcoeng.ac.in

## 1. INTRODUCTION

Image steganalysis is the art of uncovering the presence of secret in a mundane image. The digital era has provided numerous freeware steganography tools online that help novice users to embed data easily without any prior knowledge of steganographic algorithms [1]-[3]. This makes masked communication as a piece of cake to even an obvious illicit user. A simple analysis on existing steganography tools reveals the fact that most of them use lossless 24-bit image formats. The common among them are BMP, GIF, PNG and TIFF formats. Among them, BMP image format is the most widely used, because it provides a large area of hiding (implying large payload) with less probability of detection (less pixel change rate) in spite of its uncompressed data. The lossy JPEG images are least preferred because these image types are easily distorted (low payload; high pixel change rate) and detection is therefore much simpler. Thus, in general steganalysis process depends on cover image format and payload.

Most of the work carried out in the literature concentrates on the finding the artefacts produced by the

steganographic algorithm on the cover image data as a result of embedding. Universal steganographic software tool identification is scarcely reported in current literature. This is because every steganographic software tool has an underlying algorithm and detection of algorithm is sufficient for the detection of cover or stego images. However, a good deal of steganographic software tool uses the least significant bit (LSB) encoding though it is simple for digital image steganography. This makes the steganographic software tool identification as a challenging one, since the method of steganalysis of algorithm found in literature cannot be used here [4]-[15].

A wide range of simple targeted steganographic software tool identification is reported in literature. The pioneer work in the field reported that some tools leave a signature which can be exploited to identify the tool and stego-image [16]. They proved it for a set of steganographic software tools (S-Tools, Syscop, MandelSteg, etc) which used palette and fractal images. Westfeld and Pfitzmann [17] used the tools like EzStego v2.0b3, Jsteg v4, Steganos v1.5, and S-Tools v4.0 for detection by statistical steganalysis. Provos and Honeyman [18] later developed StegDetect to identify steganographic content and StegBreak to launch dictionary attack on those and retrieve the hidden content in JPEG images. Geetha *et al*. [19] identified watermarking and steganographic tools using an genetic-X-means classifier. Verma *et al*. [20] proposed a steganalysis technique on the basis of statistical observations on difference image histograms (DIH) for the reliable detection of classical least significant bit (LSB) steganography which measured the weak correlation between successive bit planes to construct a classifier for discrimination between stego-images and cover images. Sloan and Hernandez-Castro [21] reported the identification of openpuff in video steganalysis. All these were targeted (specific to single tool) and required patient scrutiny of the images manually which was time consuming and error prone.

An important work in this field was by [22] where they designed a fully automated, blind, media-type agnostic approach to steganalysis by bitwise analysis of header data and generated a signature for each tool. Though they reported their work to be media-type agnostic, the results reported were based on seven tools out of which five were of JPEG format, one in MP3 format and one in GIF format tools and a minimum of 10 stego images of each tool were used to generate the signature for the tool. This provided an insight into working out a universal steganalyser for steganographic software tool identification exploiting both artefacts in the image data and its metadata. Almost all current steganalysers require at least a little knowledge of the used steganographic software tools. Feeding the information may be a mammoth task comparing the number of tools available [23]. So a payload independent universal steganalyser is proposed that initially exploits the macroscopically changing fields of the image metadata and information from metadata to identify the tools by clustering. This helps in segregating most of the tools, while those similar to cover are further processed. This is done by forming a signature from stego images of those tools from artefacts present in their image data. The scope of the proposed universal steganalyser is limited to stego tools that work in these five image formats namely BMP, GIF, PNG, TIFF and JPEG.

The structure of this paper is as follows : Section 2. describes the structure of the Universal steganalyser; Section 3. presents the steganalysis using image header; Section 4. extends the steganalysis by comparing the signature generated for the steganographic software tool in image data; Section 5. measures and evaluates the technique experimentally; Section 6. concludes the work, and is followed by appendices and references.

## 2. PROPOSED UNIVERSAL STEGANALYSIS

The steganalysis of stego images from different steganographic software tools is done in two phases. In the first phase, the stego images are first distinguished based on one of the five image formats. Then, for each image format, certain fields or information from the fields of the header data are extracted. These features are then subjected to unsupervised clustering by means of extended K-Means. Extended K-Means clustering acts as pattern matching template to identify different steganographic software tools uniquely. Though this initial clustering identifies most of the steganographic software tools, it leaves space for steganalysis of steganographic software tools that take care of not disturbing the metadata while processing. The stego images from these steganographic software tools resemble cover images and are placed in the cluster as that of the cover. The second phase of steganalysis starts by taking these clusters. As a prerequisite for this phase, a signature is generated for each tool from the artefacts in image data. This signature is compared against the signature found in the stego images of the cluster. If a signature match is found, then the tool is identified. The block diagram of the proposed steganalyser is given in Figure 1.

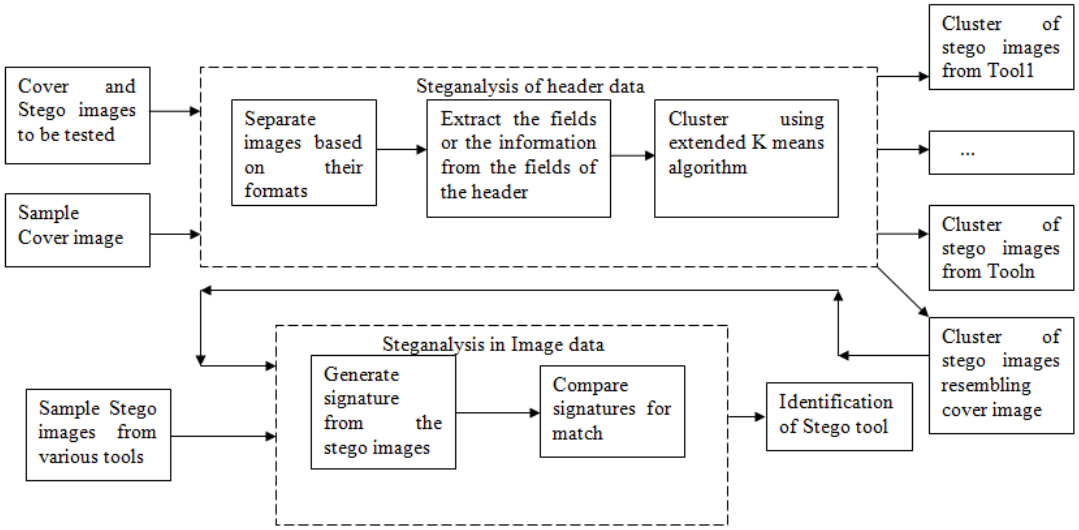



Figure 1. Block diagram of the proposed steganalyser

## 3. STEGANALYSIS USING IMAGE HEADER ARTEFACTS

A great portion of literature in digital image forensic exploit header data for various purposes [24]. Here it is used for steganalysis of steganographic software tools.

### 3.1. Fields considered in each image format

As mentioned earlier the steganalyser is to exploit the vulnerable fields of image header to identify the tool. The fields that may lead to identification of the tools are detailed for each format [25]-[29].

#### 3.1.1. BMP image format

The BMP images have a fixed byte format. The fields-bits per pixel, image data padding (last two bytes of 4 bytes of SizeofBitmap field) horizontal resolution, Vertical resolution is used since most steganographic software tool modify them. In addition, the actual size of BMP file derived from the fields is used. Thus, these five fields form features that are used for identifying tool in BMP images.

#### 3.1.2. GIF image format

There are two version formats in GIF; 87a and 89a. The trailer field is used to find camouflage steganographic software tools that do not make a single change in image but insert the secret data after the image data. The version field in file header, the packed field of the global colour table in logical screen descriptor and the packed field of the local image descriptor which has the image and colour table data information are also used. Presence of graphic control, comment and plain text extension block, size of global and local colour table are also unique features to identify tool. Thus, these fields form the features to identify tools in GIF format.

#### 3.1.3. JPEG image format

The JPEG format is dependent on the quality factor of the JPEG compression and thus can be used to distinguish not only tools but also algorithms. The fields that are exploited are as follows: JFIF version, density unit field in JFIF header, presence of data after last end of image (EOI) marker, presence of comment marker (COM), quantisation table length and location of Huffman table. These six information from header form the JPEG feature vector.

### 3.2. PNG image format

The PNG file format supports a number of chunks which help in tool identification. The presence of auxiliary chunk types like time-time of last modification, text-extensions and their cyclic redundancy check (CRC) are used to cluster tools. End of file is checked with IEND chunk field. Thus, the feature vector for PNG format is taken from these fields.

#### 3.2.1. TIFF image format

The TIFF is supported by data in two ordering: little endian and big endian, which forms the first feature. Here again presence of additional tags like artist, copyright, hostcomputer, make, model, software or datetime indicate a tool. Presence of New SubFileType or SubFileType Tag can account for significant tool identification. In addition, the fields like Number of tags in image file directory, Number of StripOffset, and information derived from RowsPerStrip, StripOffsets, StripByteCounts and DataType fields to indicate data embedded at End of image are exploited as features for TIFF images.

### 3.3. Clustering algorithm

When the labels of the given data are unknown, unsupervised learning takes place through clustering. One simple form of clustering the given information is K-Means clustering. This clustering requires number of cluster (K) as input. The provision of number of clusters is not possible in a practical scenario. So an extension to the K-Means is made by repeating with the K-Means algorithm with increasing cluster numbers until the distance of each sample to its centroid is zero. Thus, the optimal number of clusters is determined. The pseudo code for the algorithm is given as below:

Algorithm extended KMeans
Input - Features and Number of Samples
Output - Optimized Number of clusters COUNT and the Clusters, CLUST
FOR COUNT = 1 to Number of samples

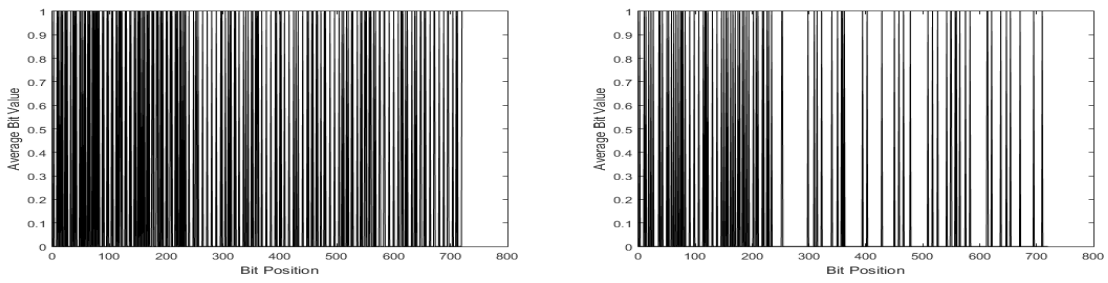
- Let Kmeans clustering of COUNT clusters based on city block distance with 5 cross validation be CLUST
- Let the within cluster distance of each cluster in CLUST be DIST
- IF DIST is equal to zero for all clusters
  - The optimal clusters and their number are found in CLUST and COUNT respectively; EXIT
- ELSE Continue

Within cluster distance of cluster is zero means exact match. Thus, the algorithm helps in identifying the exact tool's header signature which is later correlated with the tool.

## 4. STEGANALYSIS BY ARTEFACTS IN IMAGE DATA

The stego images of the steganographic software tool that do not modify the header or the metadata of the cover image cannot be detected by the above process. In order to identify those stego images and to ultimately reveal the tool, the artefacts left by the steganographic software tool in image data is considered. The metadata (stego key) about the steganographic process is in some way hidden inside the image data [16].

The fact is that the metadata is either hidden sequentially in the start or at the end of the image file or randomly. Even though the metadata may vary in byte level, it is found that at bit level, things do not change [22]. Things may be either the bit or its position. This is generated as a signature of the tool by examining its stego image data in bit level. The signature is generated from either first 100 pixels or the last 100 pixels depending on the tool and saved in signature library. This signature is compared with the bits of the stego image to be tested. A match implies the tool being used. The characteristic signature of WB Stego tool as an average of last 30 pixels over 50 images is shown in Figure 2.




(a) (b)

Figure 2. Average signature found in the last 30 pixels of 50 randomly selected BMP images, (a) WB stego BMP images, (b) Cover BMP images

Thus an automated approach for universal steganalysis of software tool is done by tracing the artefact left by the tool in both the image header and its data.

## 5. EXPERIMENTAL RESULTS AND DISCUSSION

No benchmark steganographic tools exists for steganalysis. So to create a repository of stego images, steganographic software tools are downloaded from sites referred in [23] using images from sources Bossbase, McGill databases. Table 1 lists the different steganographic software tools used.

Table 1. List of steganographic software tools used with supporting image format

| SNo | Steganographic software tool (version) | Supporting image formats |
|---|---|---|
| 1 | 2pix v1.1 (2P) | BMP |
| 2 | BlindSide v0.9b (BS) | BMP |
| 3 | DeEgger Embedder v1.3.6 (DE) | BMP, GIF, JPEG, PNG, TIFF |
| 4 | F5 (F) | JPEG |
| 5 | File in File (Fi) | BMP |
| 6 | Gifshuffle (GS) | GIF |
| 7 | Hermatic system StegPNG trail version 11.01 (hs) | BMP, PNG |
| 8 | Hide & Reveal v1.7.0 (hi) | BMP, PNG, TIFF |
| 9 | Hide in Picture v2.1 (HI) | BMP,GIF |
| 10 | Image Hide v2.0 (IH) | BMP, PNG |
| 11 | Image Protector v3.6 (IP) | BMP |
| 12 | Invisible Secrets trail version4.0 (IS) | BMP, JPEG, PNG |
| 13 | JHide v1.0.0 (JH) | BMP, PNG, TIFF |
| 14 | JPHSwin V0.5 (JP) | JPEG |
| 15 | nsF5 (ns) | JPEG |
| 16 | Open Puff v4.00 (OP) | BMP, JPEG, PNG |
| 17 | Our Secret v2.5 (OS) | BMP, GIF, JPEG, PNG, TIFF |
| 18 | Outguess v0.2 (O) | JPEG |
| 19 | Secret Layer v.2.8.1 (SL) | BMP,JPEG, PNG |
| 20 | Silent Eye v0.4.1 (SE) | BMP, JPEG |
| 21 | SSuite Piscel (SP) | BMP, PNG |
| 22 | Steganofile v1.0 (SF) | BMP |
| 23 | Steghide v0.5-win32 (sh) | BMP, JPEG |
| 24 | S-Tools v4.0 (ST) | BMP,GIF |
| 25 | Steganography Tool (imgAuthServer) (I) | PNG |
| 26 | The Secret Code Breaker Steganography program v1.2 (5) | BMP |
| 27 | Third Eye v1.0 (TE) | BMP, GIF |
| 28 | Trojan v1.0 (T) | BMP, PNG, TIFF |
| 29 | Veneer (V) | BMP, GIF, JPEG, PNG, TIFF |
| 30 | Wb - Stego v4.3 (WS) | BMP |
| 31 | Xiao v2.6.1 (X) | BMP |

For cover images, both images from clean source and internet are exploited. McGill Image database [30] which proves a challenging cover source for steganalysis is taken for clean images. Thus, the cover image database consists of 1000 images with random 500 from each source. They are basically either tiff or bmp format images. They are resized to 512 $\times$ 512 for simplicity. The cover images are then converted to five image formats namely BMP, TIFF, PNG, GIF and JPEG. For JPEG images, 100% compression ratio is used. A random 100 images from the cover source is chosen for each steganographic software tool to make the stego images for each format. Thus, a total of 6,500 (25$\times$100 BMP, 7$\times$100 GIF, 12$\times$100 JPEG, 13$\times$100 PNG, 6$\times$100 TIFF) stego images are created. The secret data is random data ranging from 1 byte to maximum possible payload by the tool. 100 random cover images (CO) for each format is also taken (though a single cover image is enough). The experiment is carried out on the set up database. The results of first phase of steganalysis are shown in Figure 3.

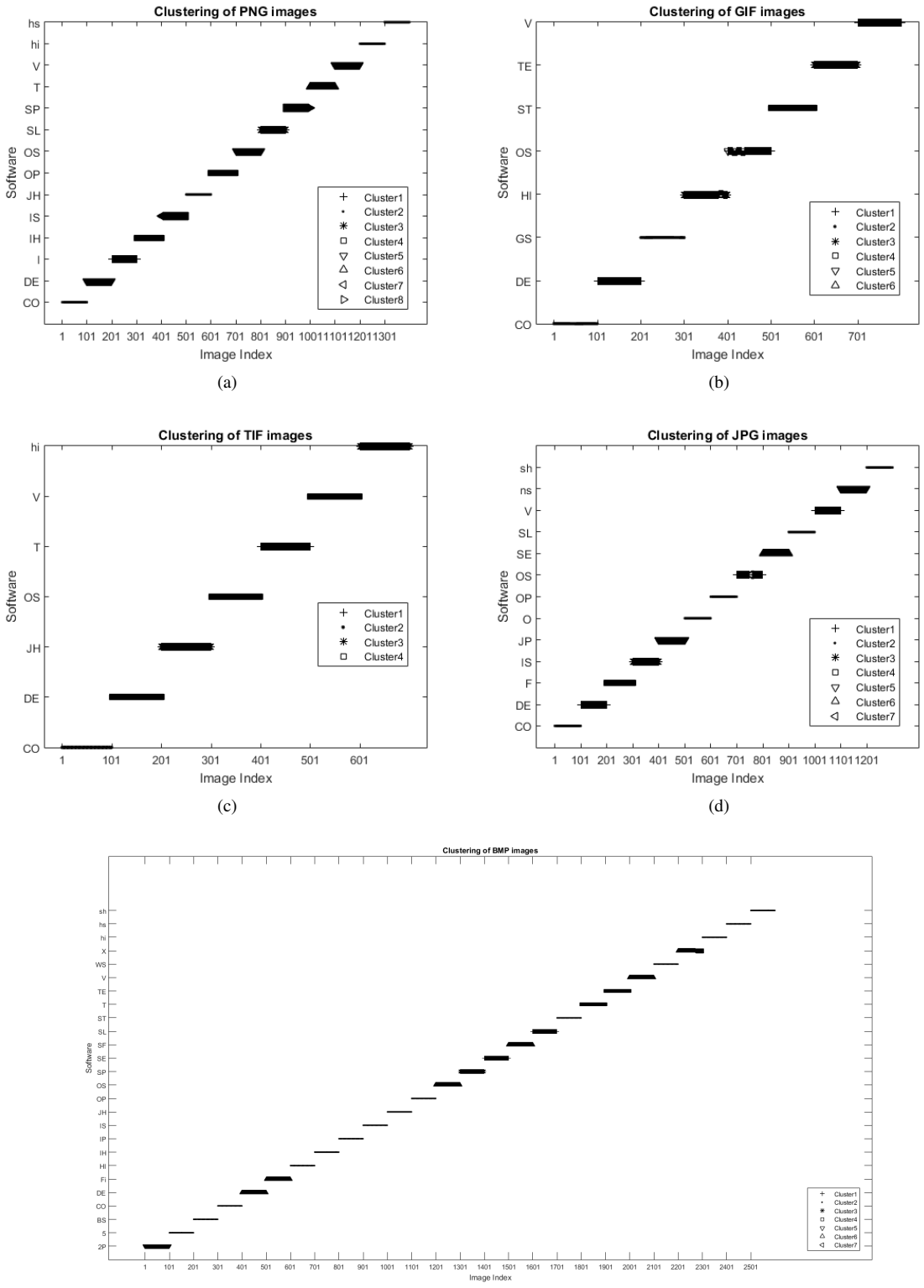



Figure 3. Scatter plot of clustering information from header data in various image formats, (a) PNG format, (b) GIF format, (c) TIFF format, (d) JPEG format, (e) BMP format

In this phase, it is noted that the steganographic software tool (DE,V,OS) that hide data at the end of image file are all clustered in separate group and are identified regardless of formats. Software specific to format (GS, IP, WB, BS) are largely difficult to identify, since care is taken by tool to leave no trace in header data. In steganographic software tool that support more than one format, at least one format is insecure (HR, HI, IS, ST, JH, OP, SL). Only one system (hs) that supports multiple format is not detectable in any of the formats. Also, it is verified that of all formats, identification of tool is very difficult in BMP because of its simple and short header, other formats have large information in header which in turn leads to loopholes or vulnerability. For the second phase, those images that resemble cover (stego images clustered along with cover) are fed as the input to the steganalyser and the results are tabulated as in Table 2.

It can be seen from Table 2 that almost all steganographic software tools store their metadata in the image data. And the signature of each tool is unique with no two tools having same signature. Also, this signature is present independent of image format. However, this phenomenon was absent in GIF format which may likely be due to the fact that GIF formats alter palette rather than data. Rarely some tools like BS, HI and ST handle it. Either they do not store meta data in its image or they are stored randomly.

Table 2. Experimental results for signature matching in image data of different steganalyser tools

| S.No | Image format | Tool | Signature length (in bits) and location | Average signature match in % | |
|---|---|---|---|---|---|
| | | | | True positive | False positive |
| 1 | BMP | 5 | 14 Top | 100 | 0 |
| 2 | | BS | No signature | 0 | 0 |
| 3 | | HI | No signature | 0 | 0 |
| 4 | | IH | 145 Top | 100 | 0 |
| 5 | | IP | 83 Top | 100 | 0 |
| 6 | | IS | 101 Top | 100 | 0 |
| 7 | | JH | 58 Bottom | 100 | 0 |
| 8 | | OP | 24 Bottom | 100 | 0 |
| 9 | | ST | No signature | 0 | 0 |
| 10 | | WS | 284 Bottom | 100 | 0 |
| 11 | | hi | 260 Top | 100 | 0 |
| 12 | | hs | 39 Bottom | 100 | 0 |
| 13 | | sh | 132 Bottom | 100 | 0.7 |
| 14 | GIF | GS | | 100 | |
| 15 | JPEG | O | 60 Bottom | 100 | 0 |
| 16 | | OP | 72 Bottom | 100 | 0.25 |
| 17 | | SL | 67 Bottom | 100 | 0.25 |
| 18 | | sh | 99 Bottom | 100 | 0.75 |
| 19 | PNG | JH | 60 Bottom | 100 | 0 |
| 20 | | hi | 173 Top | 100 | 0 |
| 21 | | hs | 20 Bottom | 100 | 0 |

Comparison of this steganalyser is done with existing freeware steganalyser namely StegSpy v2.1 [31], StegSecret also known as XStegSecretBeta v0.1 [32], StegExpose [33]. StegSpy claims to identify the following programs: Hiderman, hide and seek, masker, JPegX and Invisible Secrets. Following are the techniques and tools identified by StegSecret (PNG & TIFF formats not supported) : Tools - Camouflage V1.2.1, inThePicture v2, JPEGXv2.1.1, pretty good envelope PGE) v1.0, appendX v less than 4, Steganography v1.6.5, inPlainView, DataStash v1.5 and dataStealth v1.0. Techniques - EOF techniques, Visual Attacks, ChiSquare Attack and RS Attack. Sequential and Pseudorandom LSBs, uses BDAS v0.1 (steganography tools fingerprint DataBase) to detect more than 40 steganography tools. StegExpose (TIFF format not supported) is statistical tool for identifying LSB embedding in images. It employs ChiSquare attack, RS attack and primary set attacks to identify tool. These tools are used against generic cover, stego classification on the set up database. The results are noted in Table 3.

Table 3. Percentage of stego images identified by different steganalyser tools

| Stegano graphic tool | Steg expose | Steg secret | Steg spy | Proposed steganalyser |
|---|---|---|---|---|
| BMP image format | | | | |
| 2P | 30 | 0 | 30 | 100 |
| BS | 0 | 0 | 5 | 0 |
| DE | 20 | 100 | 50 | 100 |
| Fi | 10 | 100 | 90 | 100 |
| hs | 100 | 0 | 90 | 100 |
| hi | 0 | 0 | 60 | 100 |
| HI | 20 | 0 | 90 | 0 |
| IH | 10 | 0 | 70 | 100 |
| IS | 100 | 0 | 30 | 100 |
| IP | 10 | 0 | 60 | 100 |
| JH | 0 | 0 | 60 | 100 |
| OP | 70 | 0 | 50 | 100 |
| OS | 10 | 100 | 30 | 100 |
| SL | 40 | 0 | 40 | 100 |
| SE | 40 | 0 | 50 | 100 |
| SP | 0 | 0 | 40 | 100 |
| SF | 0 | 100 | 70 | 100 |
| sh | 0 | 0 | 50 | 100 |
| ST | 0 | 0 | 40 | 0 |
| 5 | 0 | 0 | 60 | 100 |
| TE | 0 | 10 | 50 | 100 |
| T | 10 | 0 | 70 | 100 |
| V | 0 | 100 | 50 | 100 |
| WS | 10 | 0 | 40 | 100 |
| X | 0 | 70 | 40 | 100 |
| GIF image format | | | | |
| DE | 0 | 0 | 10 | 100 |
| GS | 0 | 0 | 0 | 0 |
| HI | 0 | 0 | 0 | 100 |
| OS | 10 | 100 | 20 | 100 |
| ST | 0 | 0 | 0 | 100 |
| TE | 0 | 0 | 40 | 100 |
| V | 10 | 100 | 20 | 100 |

| Stegano graphic tool | Steg expose | Steg secret | Steg spy | Proposed steganalyser |
|---|---|---|---|---|
| JPEG Image format | | | | |
| DE | 10 | 100 | 10 | 100 |
| F5 | 0 | 0 | 0 | 100 |
| IS | 10 | 0 | 10 | 100 |
| JP | 20 | 0 | 0 | 100 |
| ns | 10 | 0 | 0 | 100 |
| OP | 0 | 0 | 10 | 100 |
| OS | 10 | 100 | 0 | 100 |
| O | 10 | 0 | 0 | 100 |
| SL | 20 | 0 | 10 | 100 |
| SE | 0 | 0 | 0 | 100 |
| sh | 10 | 0 | 10 | 100 |
| V | 0 | 100 | 0 | 100 |
| PNG image format | | | | |
| DE | 0 | 0 | 50 | 100 |
| hs | 100 | 0 | 10 | 100 |
| hi | 60 | 0 | 20 | 100 |
| IH | 0 | 0 | 20 | 100 |
| IS | 0 | 0 | 30 | 100 |
| JH | 0 | 0 | 20 | 100 |
| OP | 0 | 0 | 40 | 100 |
| OS | 0 | 0 | 10 | 100 |
| SL | 50 | 0 | 40 | 100 |
| SP | 50 | 0 | 10 | 100 |
| I | 10 | 0 | 0 | 100 |
| T | 0 | 0 | 0 | 100 |
| V | 0 | 0 | 60 | 100 |
| TIFF image format | | | | |
| DE | 0 | 0 | 70 | 100 |
| hi | 0 | 0 | 70 | 100 |
| JH | 0 | 0 | 70 | 100 |
| OS | 0 | 0 | 80 | 100 |
| T | 0 | 0 | 50 | 100 |
| V | 0 | 0 | 70 | 100 |

## 6. CONCLUSION

In almost all the formats, the identification of tool by its stego image independent of payload is a major contribution of the proposed steganalyser over the statistical steganalyser which finds detection of stego images with payload less than 5% of the maximum capacity as an arduous task. The other facts that can be concluded from the experimentation are almost all tools leave a trace in either the header or the data of the stego image. BMP format has the least vulnerable header of all image formats and size of secret payload is irrelevant because it is not related to the image statistics but to the tool signature. Thus, this universal blind structural steganalyser is capable of identifying tools which leave their trace in the stego images irrespective of the size of secret payload. Conversely, this means that this method will not operate at all against implementations of algorithms that do not produce characteristic irregularities in their header (simple LSB batchwise processing) or store their metadata in the image BS. However, at large, a match against a stego signature can provide a useful indication that a particular tool may have been used and consequently an indication that the file may contain steganography. Also, this steganalyser produces a 100% match for a tool irrespective of payload which cannot be the case for other statistical steganalyser. So such universal structural steganalysers can effectively be deployed as the pre mechanisms to the existing steganalysis techniques and help to improve overall accuracy.

## REFERENCES

[1] N. F. Johnson and S. Jajodia, "Exploring steganography: Seeing the unseen," in *Computer*, vol. 31, no. 2, pp. 26-34, Feb. 1998, doi: 10.1109/MC.1998.4655281.

[2] "Steganographyonline", [Online]. Available: https://stylesuxx.github.io/steganography/.

[3] "Manytools", [Online]. Available: https://manytools.org/hacker-tools/steganography-encode-text-into-image/.
[4] J. Fridrich, M. Goljan and R. Du, "Reliable detection of LSB steganography in color and grayscale images," *MM&Sec '01: Proceedings of the 2001 workshop on Multimedia and security: new challenges*, 2001, pp. 27-30, doi: 10.1145/1232454.1232466.
[5] J. Fridrich, M. Goljan and Rui Du, "Detecting LSB steganography in color, and gray-scale images," in *IEEE MultiMedia*, vol. 8, no. 4, pp. 22-28, Oct.-Dec. 2001, doi: 10.1109/93.959097.
[6] A. Westfeld, "F5—A Steganographic Algorithm," in *Information Hiding*, vol. 2137, 2001, doi: 10.1007/3-540-45496-9_21.
[7] Tao Zhang and Xijian Ping, "Reliable detection of LSB steganography based on the difference image histogram," *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, 2003, pp. III-545, doi: 10.1109/ICASSP.2003.1199532.
[8] S. Dumitrescu, Xiaolin Wu and Zhe Wang, "Detection of LSB steganography via sample pair analysis," in *IEEE Transactions on Signal Processing*, vol. 51, no. 7, pp. 1995-2007, July 2003, doi: 10.1109/TSP.2003.812753.
[9] A. D. Ker, "Steganalysis of LSB matching in grayscale images," in *IEEE Signal Processing Letters*, vol. 12, no. 6, pp. 441-444, June 2005, doi: 10.1109/LSP.2005.847889.
[10] S. Lyu and H. Farid, "Steganalysis using higher-order image statistics," in *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 111-119, March 2006, doi: 10.1109/TIFS.2005.863485.
[11] B. Li, J. Huang and Y. Q. Shi, "Steganalysis of YASS," in *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 3, pp. 369-382, Sept. 2009, doi: 10.1109/TIFS.2009.2025841.
[12] J. Fridrich, J. Kodovský, V. Holub, and M. Goljan, "Breaking HUGO – The Process Discovery," in *Information Hiding*, vol. 6985, 2011, doi: 10.1007/978-3-642-24178-9_7.
[13] K. Karampidis, E. Kavallieratou and G. Papadourakis, "A review of image steganalysis techniques for digital forensics," *Journal of Information Security and Applications*, vol. 40, 2018, doi: 10.1016/j.jisa.2018.04.005.
[14] S. Chutani and A. Goyal, "A review of forensic approaches to digital image Steganalysis," *Multimedia Tools and Applications*, vol. 78, pp. 18169–18204, 2014, doi: 10.1007/s11042-019-7217-0.
[15] O. I. Al-Sanjary, O. Ahmed Ibrahim and K. Sathasivem, "A New Approach to Optimum Steganographic Algorithm for Secure Image," *2020 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, 2020, pp. 97-102, doi: 10.1109/I2CACIS49202.2020.9140186.
[16] N. F. Johnson and S. Jajodia, "Steganalysis of Images Created Using Current Steganography Software," in *Information Hiding*, vol. 1525, 1998, doi: 10.1007/3-540-49380-8_19.
[17] A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems," in *Information Hiding*, vol. 1768, 2000, doi: 10.1007/10719724_5.
[18] N. Provos and P. Honeyman, "Detecting Steganographic Content on the Internet," *NDSS Symposium 2002*, 2002.
[19] S. Geetha, S. S. Sindhu and N. Kamaraj, "Detection of stego anomalies in images exploiting the content independent statistical footprints of the steganograms," *informatica*, vol. 33, no. 1, 2009.
[20] S. Verma, S. Sood and S. K. Ranade, "Relevance of Steganalysis using DIH on LSB Steganography," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 2, pp. 835-838, 2014.
[21] T. Sloan and J. Hernandez-Castro, "Steganalysis of OpenPuff through atomic concatenation of MP4 flags," *Digital Investigation*, vol. 13, pp. 15-21, 2015, doi: 10.1016/j.diin.2015.02.002.
[22] G. Bell and Y. Lee, "A Method for Automatic Identification of Signatures of Steganography Software," in *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 354-358, June 2010, doi: 10.1109/TIFS.2010.2046985.
[23] "Steganography Software," 2015. [Online]. Available: http://www.jjtc.com/Steganography/tools.html.
[24] P. Alvarez, "Using Extended File Information (EXIF) File Headers in Digital Evidence Analysis," *International Journal of Digital Evidence*, vol. 2, no. 3, pp. 1-5, 2004.
[25] H. S. Hussain, R. Din, M. H. Ali, M. H. and N. Balqis, "'The Embedding Performance of StegSVM Model in Image Steganography," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 12, no. 1, pp. 233-238, 2018, doi: 10.11591/ijeecs.v12.i1.pp233-238.
[26] B. A. Sultan and L. E. George, "Color image compression based on spatial and magnitude signal de-

composition," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 5, pp. 4069-4081, 2021, doi: 10.11591/ijece.v11i5.pp4069-4081.
[27] w3consortium, "PNG Specification: File Structure," [Online]. Available: http://www.w3.org/TR/PNG-Structure.html.
[28] w3consortium, ""GIF87a Specification," [Online]. Available: http://www.w3.org/Graphics/GIF/spec-gif87.txt.
[29] w3consortium, "GIF89a Specification," [Online]. Available: http://www.w3.org/Graphics/GIF/spec-gif89a.txt.
[30] "Original McGill Colour Image Database," [Online]. Available: http://tabby.vision.mcgill.ca/.
[31] "SpyHunter," [Online]. Available: http://www.spy-hunter.com/stegspydownload.htm.
[32] "StegSecret. A simple steganalysis tool," [Online]. Available: http://stegsecret.sourceforge.net/.
[33] "StegExpose," [Online]. Available: https://github.com/b3dk7/StegExpose.