

Malicons: Detecting Payload in Favicons

Tomáš Pevný^{1,2}, Martin Kopp^{2,3}, Jakub Křoustek⁴, Andrew D. Ker⁵

¹Department of Computer Science, Czech Technical University in Prague, Czech Republic

²Cisco R&D Center in Prague, Cisco Systems, Inc.

³Faculty of Information Technology, Czech Technical University in Prague, Czech Republic

⁴AVG Technologies, Inc., Czech Republic

⁵Department of Computer Science, Oxford University, United Kingdom

Abstract

A recent version of the “Vawtrak” malware used steganography to hide the addresses of the command and control channels in favicons: small images automatically downloaded by the web browser. Since almost all research in steganalysis focuses on natural images, we study how well these methods can detect secret messages in favicons. The study is performed on a large corpus of favicons downloaded from the internet and applies a number of state-of-art steganalysis techniques, as well as proposing very simple novel features that exploit flat areas in favicons. The ultimate question is whether we can detect Vawtrak’s steganographic favicons with a sufficiently low false positive rate.

Motivation

The term *botnet* refers to a group of computers infected by the same malicious software with a communication channel to the *bot-master*, which can instruct the infected computers (further called *bots*) to execute various tasks that make profit for the bot-master. Examples of such tasks include sending spam e-mails, participating in distributed denial of service attacks, collecting banking or other login credentials [13, 14], copying credit card numbers, stealing business or private documents and exfiltrating them to the bot-master, recording voice or video through connected capturing devices, or using bots as a proxy or relay to conceal bot-master’s illegal operations. Botnets are therefore dangerous, as they not only directly damage the user of the infected computer, but also can make them participants in illegal activity. Operating botnets is now multi-billion dollar [18, 21] industry, with dedicated programmers refining the software to make infection as invisible as possible and evade detection.

The detection of bots and botnets is done on many fronts. The traditional approach, taken by antivirus systems, is to search for a sequence of bytes unique to known threats. The effectiveness of this solution is decreasing, since malware authors employ evasion techniques such as polymorphism and encryption. Complementary methods focus on identification of behaviour characteristics of botnets, either using logs provided by operating systems or extensions, or using logs of network devices such as routers or proxy servers. The latter is particularly interesting, since installation of other software is not needed on individual computers and the detection system has a global view of the protected system.

The botnet has a value to its bot-master if bots and bot-master can communicate with each other. Therefore many intrusion detection systems focus on detecting this communication link called the *command and control (C&C) channel*. Consequently malware authors try to make this communication as in-

visible as possible, and adoption of stenographic techniques is a logical step [20, 13, 15]. The use of stenography in botnets does not need to be limited to avoiding detection of the C&C channels: it can be used to conceal exfiltration of stolen data [22] or to download the initial bot’s configuration, for example with addresses of the C&C servers.

The motivation for this work comes from a recent report [13] about a banking trojan called Vawtrak.¹ Vawtrak has sophisticated functionality: it disables installed antivirus, can steal credentials, records voice and video, provides remote access to the desktop of infected computers, and acts as a SOCKS proxy. Moreover, Vawtrak regularly checks for updates and can be extended by downloading modules, adding new functionality. Vawtrak encrypts all its communication with C&C servers by a variation of the one-time pad with a pseudo-randomly generated key. The list of C&C servers is regularly updated by downloading a *favicon* (a small image automatically downloaded by a web browser, to be displayed in the left part of the address bar) from the last known C&C servers, and the new addresses of C&C servers are hidden inside using a *steganographic algorithm*. Using favicons is ingenious, because their download rarely raises suspicion: they are commonly download and their content is rarely inspected. Moreover, Vawtrak downloads them only during the user’s browsing activity.

This work experimentally assesses the capability of state of the art steganalysis to detect Vawtrak’s stego favicons. We will see that its embedding method is naive and unsophisticated, but it is not obvious that the methods from the steganographic literature will work well, since they focus on natural images (usually represented by BossBase [2]) and the methods may be overfitted to this type of image rather than icons, and to the adaptive steganography that is nowadays targeted by the literature. Furthermore, their sophistication may be unnecessary against simple embedding such as Vawtrak’s. The paper demonstrates this by showing a very simple set of features with negligible computational complexity, but surprisingly good accuracy on favicons. The experimental section also considers structural detectors, to investigate if they have advantage on noiseless images.

This paper is organized as follows. The next section provides necessary details on how Vawtrak uses steganography, and speculates about its possible embedding algorithm. We follow with a brief survey of the steganalysis used in this study, and introduce simple features tailored to noiseless images. The experiments sec-

¹Banking trojans are primarily designed to steal the user’s login credentials for their bank.

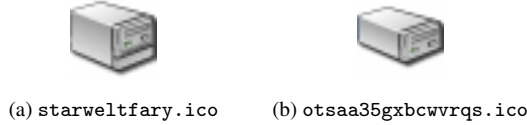


Figure 1: Favicons with hidden messages used by Vawtrak.

```

ff fe ff ff ff fe ff ff    ff ff fe fe ff ff fe fe
fe fe ff fe fe ff ff fe    fe fe fe fe fe fe ff ff
ff fe fe ff ff fe fe ff    ff fe fe fe ff fe ff fe
ff ff ff fe fe fe fe fe    fe fe ff ff fe ff ff fe
ff ff ff fe fe d7 ac ad    fe ff fe ff fe fe ff fe
fe ea c2 af ae d7 fb fa    fe fe ff fe fe ff fe ff
a5 b8 f2 fc fa fb f9 f7    b9 eb ff ff fe fe fe fe
9d b8 cc eb f2 f7 f4 f5    a9 80 76 86 c2 f9 ff fe

fe fe ff f7 e8 c9 90 56    c5 c6 90 5b 3d 51 6e 88
fe ff fe fe ff fb ef d9    74 41 42 5e 7c a3 d0 ec
ff fe fe fe fe ff ff fe    4e 6b 89 b4 e2 f2 fa fe
ff ff ff ff fe ff fe ff    98 cf eb f6 ff ff fe ff
ff fe ff ff fe ff ff ff    ee fa fe fe ff fe fe ff
ff ff ff ff fe fe fe fe    fe ff ff fe fe ff ff fe
fe ff fe fe ff fe ff ff    fe ff ff ff ff ff ff ff
fe ff ff fe fe fe ff fe    ff fe fe ff fe ff fe ff

```

Figure 2: Hexadecimal dump of the corners of starweltfary.ico's first color plane. All corners exhibit the small deviations from white caused by embedding changes.

tion compares the methods. The paper closes with a conclusion.

Steganography in Vawtrak

Vawtrak updates its list of Command and Control servers by downloading a favicon with a steganographic payload. The favicons are always downloaded during the user's surfing activity, which complicates detection: modern browsers simultaneously open multiple connections to download webpage content, possibly located on different servers belonging to different companies. Therefore downloading an unrelated favicon in the middle of a webpage download rarely raises an alert for behaviour-based intrusion detection systems. From this point of view, favicons represent a great covert channel. Vawtrak always uses true-color favicons of size 32×32 , and the two we have captured containing a hidden message are shown in Figure 1.

This study works with limited information, in that we do not know the entire mode of operation of Vawtrak. Our information comes from observation of networks on which it is active, and reverse-engineering of the malware binary (which *decodes* the payload). The extraction of an encrypted hidden message can be summarized as follows:

1. check if the size of the favicon is 4286 bytes (if not, terminate as the favicon is not targeted at Vawtrak);
2. skip the header of the favicon, the first 62 bytes;
3. extract hidden data from the least significant bit of pixels in sequential order while skipping every fourth byte (this cor-

responds to the alpha channel².) The extraction is finished after extracting 288 bytes of the hidden message.

This extracted *encrypted* message is always 288 bytes long. The first four bytes forms the cryptographic key for an RC4 cipher. The RC4 cipher [19] is used ten times on the remaining 284 extracted bytes, plus 92 bytes following them in the memory. These 92 bytes are random (they are not initialized) and they do not affect the decoding algorithm. The decoded message contains an MD5 signature in the first 16 bytes to verify that the favicon is from Vawtrak's C&C. This is presumably a protection measure against bot-hijacking, although like the cryptography it must provide poor protection in practice: a hijacker with knowledge of the algorithm could choose their own key, and replace the MD5 signature with their own.

The upper-left area of the icon starweltfary.ico should be plain white, but the hexadecimal dump of the first color plane in Figure 2 shows the variations caused by embedding algorithm. We cannot identify the embedding algorithm in Vawtrak's binary, because Vawtrak uses the steganography only in the direction from the bot-master to infected bot; from Google's image search we were able to find the original cover icons, but unfortunately they were in other format than .ico, which prevented direct knowledge of the embedding changes. Nevertheless the extraction procedure reveals that

1. the steganographic algorithm is non-adaptive,
2. it is not protected by a stego-key randomizing the location of embedding bits,
3. and the message is hidden either by Least Significant Bit Replacement (LSBR) or Matching (LSBM).

The first two properties come from the fact that the message is read sequentially without any syndrome coding. The third property comes from the fact that the message is read from the least significant bit. A close inspection of the two seized stego favicons further reveals that, although the extraction algorithm extracts only 288 bytes, the least significant bitplanes of all three color channels of the entire image have been modified: all corners which should have been plain white have embedding artefacts, as seen in Figure 2. In the experiments below, we have therefore assumed that Vawtrak's favicons were fully embedded using LSBR or LSBM.

Favicons as steganographic media

Favicons are small images named favicon.ico (regardless of their format), which a web browser automatically tries to download when a webpage is visited. If the favicon is successfully downloaded, and the format is supported, the favicon is displayed in the left part of the browser's address bar. Favicons have standardized sizes of 16×16 , 32×32 , 48×48 or 64×64 pixels with 1, 2, or 3 color planes and possibly an alpha channel. Most favicons are stored in a spatial (bitmap) format, although some modern browsers now allow others such as JPEG and SVG. Favicons are mostly computer generated graphics or icons with many flat areas, which means that they virtually lack noise. This considerably decreases the steganographic capacity, since these areas

²The .ico container internally stores image data in BMP or PNG format; in either case, each true-color pixel is stored as four adjacent bytes with the last one being the alpha channel.

should be avoided (though Vawtrak does not do this). The small size also means that there is likely to be very little spare capacity, likely allowing no scope for adaptive embedding unless the payload is tiny.

Given our information about Vawtrak, we believe that the message will probably be longer than four bytes (the bare minimum necessary for an IP address of a C&C server). Malware usually uses domain names instead of plain IP address, as they are more robust to blocking; for the same reason, there will be multiple C&C server addresses. Also, Vawtrak signs the hidden data to try to prevent botnet hijacking³. All these requirements increase the length of hidden data.

To summarize the pros and cons of favicons for implementing an innocuous channel, their advantage is that they are unlikely to be scrutinized and due to their size there is little evidence on which to build a reliable detector (due to the square root law [6] a certain relative payload is harder to detect in small covers than large). On the other hand, they have little noise and their small capacity prevents the use of advanced adaptive algorithms.

Steganographic detectors

This section first presents algorithms from prior art, and then describes a set of simple features called *Patch* that exploit the low level of noise in favicons. We will compare them in the following section. Surprisingly, the simple features achieve better accuracy than the state of the art for color images [8]. As well as accuracy they have substantially lower computational complexity, which is a frequently-ignored property and important for practical applications.

Prior art

Structural detectors are based on manually-discovered statistics that predictably change with the length of the hidden message. Their biggest advantage is the lack of a training phase, which decreases the chance of being over-fitted to the source of covers used to create a training set, or a particular length of message. On the other hand, their accuracy is frequently inferior to that of feature-based detectors. The experimental comparison here includes the following four structural detectors of LSBR: Sample Pairs (SP) [4], Triples Analysis (Triples) [10], Weighted-Stego Image (WS) [11] and Asymptotically Uniformly Most Powerful (AUMP) [5].

Feature based detectors represent an image by a large and fairly generic set of features sensitive to embedding changes yet insensitive to image content. These features are used by machine learning algorithms to learn a decision statistic to classify images as cover or stego. The main weakness of this paradigm is that the classifier is optimized to the particular combination of image source used for the training set, steganographic algorithm, and the distribution of payload in stego images of the training set. In the case of mismatch between any of these quantities, the accuracy

³Botnet hijacking refers to the situation where one bot-master steals bots controlled by a different bot-master, redirecting C&C traffic to his own servers, for example by replacing their lists of C&C servers. Since the same binaries are used by different bot-masters, and the binaries now offered in a “make your own botnet” [1] fashion, botnet hijacking is simplified and protection measures are therefore needed. Moreover, similar techniques are used by security researchers to learn details about the structure of botnets and their C&C commands.

rapidly decreases [12, 16]. From the plethora of available features for steganalysis, we have chosen the Color Rich Model (CRM) features [8], because they are designed for true-color images. The machine learning method used to train the final classifier from features is discussed in the Experiments section.

Patch features

Since favicons are computer generated, they contain very little noise and they have relatively large areas of flat colors (see, for example, the favicons used by Vawtrak in Figure 1). These areas are completely flat with no variation in the least significant bitplane of neighbouring pixels. The proposed *patch features* try to capture this flatness, and the effect of LSB steganography on flat areas. The features are a normalized histogram of patches of size 2×2 , with pixel values reduced modulo 2^q ; the parameter q controls how many least significant bits are modelled, and the number of features. Denoting pixels of, for example, the red color plane as $\{r_{i,j}\}_{1,1}^{n,m}$, then the index of one patch is calculated as

$$\text{mod}(r_{i,j},s) + \text{mod}(r_{i,j+1},s)s + \text{mod}(r_{i+1,j},s)s^2 + \text{mod}(r_{i+1,j+1},s)s^3,$$

where $s = 2^q$. In the experiments presented below, $q = 3$, which leads to 4096 features from one color plane. Calculation of the indexes of patches, and the feature vector thereof, is very simple and can be achieved in one line of MATLAB as

$$x = \text{conv2}(\text{mod}(r,s), s.^{[0,1;2,3]}, 'valid') + 1;^4$$

These *within-plane* patches are extracted from all three color planes, and the resulting histograms are averaged.

A similar approach has been used to capture dependency across the color planes, in which case the square patch is taken vertically. This has been inspired by the recent work on a Color Rich Model for steganalysis [8]. Denoting, for example, the green color plane as $\{g_{i,j}\}_{1,1}^{n,m}$, the index of one patch between red and green color planes is calculated as

$$\text{mod}(r_{i,j},s) + \text{mod}(r_{i,j+1},s)s + \text{mod}(g_{i,j},s)s^2 + \text{mod}(g_{i,j+1},s)s^3.$$

Again the calculation of indexes can be written in one MATLAB line as

$$x = \text{conv2}(\text{mod}(r,s), s.^{[0,1]}, 'valid') + \text{conv2}(\text{mod}(g,s), s.^{[2,3]}, 'valid') + 1;$$

Across-plane patches were extracted from all three combinations of red, green, and blue color planes and resulting histograms were averaged.

The final number of features is $2s^4$, which for $q = 3$ makes 8192 features.

Patch features can be extended using different neighbourhoods, as is done in rich models [7], but the goal was to keep the features simple so that their extraction would be fast. Indeed our extraction routine takes about 0.001 seconds for one favicon (on Intel Xeon clocked at 2.40GHz), whereas the extraction of CRM features takes 0.4 seconds: 400 times longer.

⁴The calculation of the histogram of indexes in MATLAB can be done as $F = \text{accumarray}(x(:), 1, [s^4, 1])$.

Experiments

Favicon and natural databases

We prepared two databases of images sized 32×32 : genuine favicons, and crops from photographic images. They will allow us to tell which steganographic techniques are stronger for noiseless favicons rather than natural images with noise.

To collect genuine favicons of the same type as Vawtrak’s, we have filtered logs from Cisco’s Cloud Web Security [9]. We identified and downloaded nearly 2 million favicons, of which 136 039 were of the required format: true-color and of size 32×32 . Because they have been collected from genuine networks, the favicon database should be representative of such images found in the real world. It has been assumed that all these favicons are clean (without hidden message): although there were a few with the size 4286 bytes used by Vawtrak, running the extraction and decoding routine on them did not yield a meaningful message.

The database of “natural” small images with noise was prepared from the raw color images in BossBase [2] (the creation had to start with raw images since the commonly-used BossBase 1.01 is in grayscale). Each image was resized to 256×256 and then 10 random crops of size 32×32 were taken from each. This lead to a total of 100 000 images.

Experimental settings

The structural detectors used default settings of parameters supplied with the reference implementations.⁵ The detectors based on CRM and Patch features were trained by linear ridge-regression, which was preferred over the popular ensemble of Fisher Linear Discriminants due to its faster training and similar accuracy [3]. The classifier was trained on 50% of images and the remaining 50% were used for testing. The regularization parameter (tolerance) was found by a line search estimating the accuracy by 5-fold cross-validation on the set of values $\{10^{-6} \cdot 2^m | m \in \{1, 2, \dots, 20\}\}$, and then using the least value for the final training. Unless stated otherwise, the error was measured by the usual probability of error under equal prior of observing cover and stego image, i.e. $P_E = \min \frac{1}{2} (P_{Fp} + P_{Fn})$, where P_{Fp}/P_{Fn} stands for the probability of false positives / false negatives.

Experimental results

Clairvoyant case

The first experiment compares the chosen detectors in the unrealistic scenario commonly assumed in the literature: the embedding algorithm and the length of possibly-hidden message are known. The error P_E of all combinations of embedding algorithm (LSBR vs. LSBM) and source of images (favicons vs. natural) is shown in Figure 3, where the payload length is varied from 0.2 bits per pixel component (77 bytes) to full embedding (384 bytes). Recall that we believe that Vawtrak is using either LSBR or LSBM at maximal length.

The structural detectors are targeted only towards LSBR, and analysis shows that they ought to be completely insensitive to LSBM. This is indeed the case in natural images (their error rate is 50%), but in fact their error rates are only around 15% for favicons embedded with LSBM. We can attribute this to the effect of fully-saturated areas which form the background of the icons: on

⁵Implementations of all detectors have been downloaded from http://dde.binghamton.edu/download/structural_lsb_detectors/.

	starweltfary	otsaa35gxbcwvrqs
WS	25	17
SP	0	0
Triples	0	0
AUMP	0	0
CRM / LSBM	440	210
CRM / LSBR	14	4
Patch / LSBM	0	0
Patch / LSBR	0	0

Table 1: Number of cover favicons classified incorrectly as stego (false positives) of structural and feature-based detectors when their operating point is set such that they just classify Vawtrak’s favicons. The cover favicons were taken from a fixed testing set that contained 68019 favicons (even for structural detectors where there was no training set). There are two variants of the feature-based detectors, one trained to detect LSBM and the other trained to detect LSBR.

such pixels, LSBR and LSBM are the same embedding operation. The AUMP detector, which is based on a model of digital images, does not perform at all well on favicons: they do not conform to its model.

The CRM and Patch features perform better (the only structural detector that could outperform them was Triples, and only in the case of LSBR in natural images) and indeed the error rates are close to zero for fully-embedded images. Using the Patch features in favicons, the error P_E is 0.015% for LSBM and 0.006% for LSBR. The Patch features achieved lower errors than the more expensive CRM features in all but three cases; the fact that they surpassed CRM even in natural images was rather surprising as they were designed primarily to take the advantage of the noiselessness of favicons.

The detectability of the two stego favicons that we know to be used by Vawtrak was measured by calculating the the number of cover favicons detected as stego (number of false positives), when the detector’s threshold is set such that Vawtrak’s favicons would be just detected as stego. For practical detectors in computer security the false positive rate is a crucial quantity due to the prevalence of cover samples [17]. The formula for the observed number of false positives is

$$P_{Fp} = \sum_{i \in \mathcal{I}_c} I(w^T(x_i - x_m) \geq 0),$$

where \mathcal{I}_c denotes the set of cover images, x_m is the feature vector of Vawtrak’s stego favicon, w is the projection vector of the linear classifier, and $I(\cdot)$ is one if its argument is true, zero otherwise. The formula for structural detectors is similar if $w = 1$ and the features x_i are outputs of the decision statistic.

The number of false positives for the clairvoyant detectors is shown in Table 1. For feature based detectors, they were trained on images with 3 bits per pixel (full embedding) and either LSBR or LSBM. These results are encouraging, since all detectors except WS and CRM detected Vawtrak’s favicons with zero false positives, which means that if state of the art steganalyzers were deployed then they would work with high accuracy. The results of this experiment also suggest that Vawtrak is more likely to have

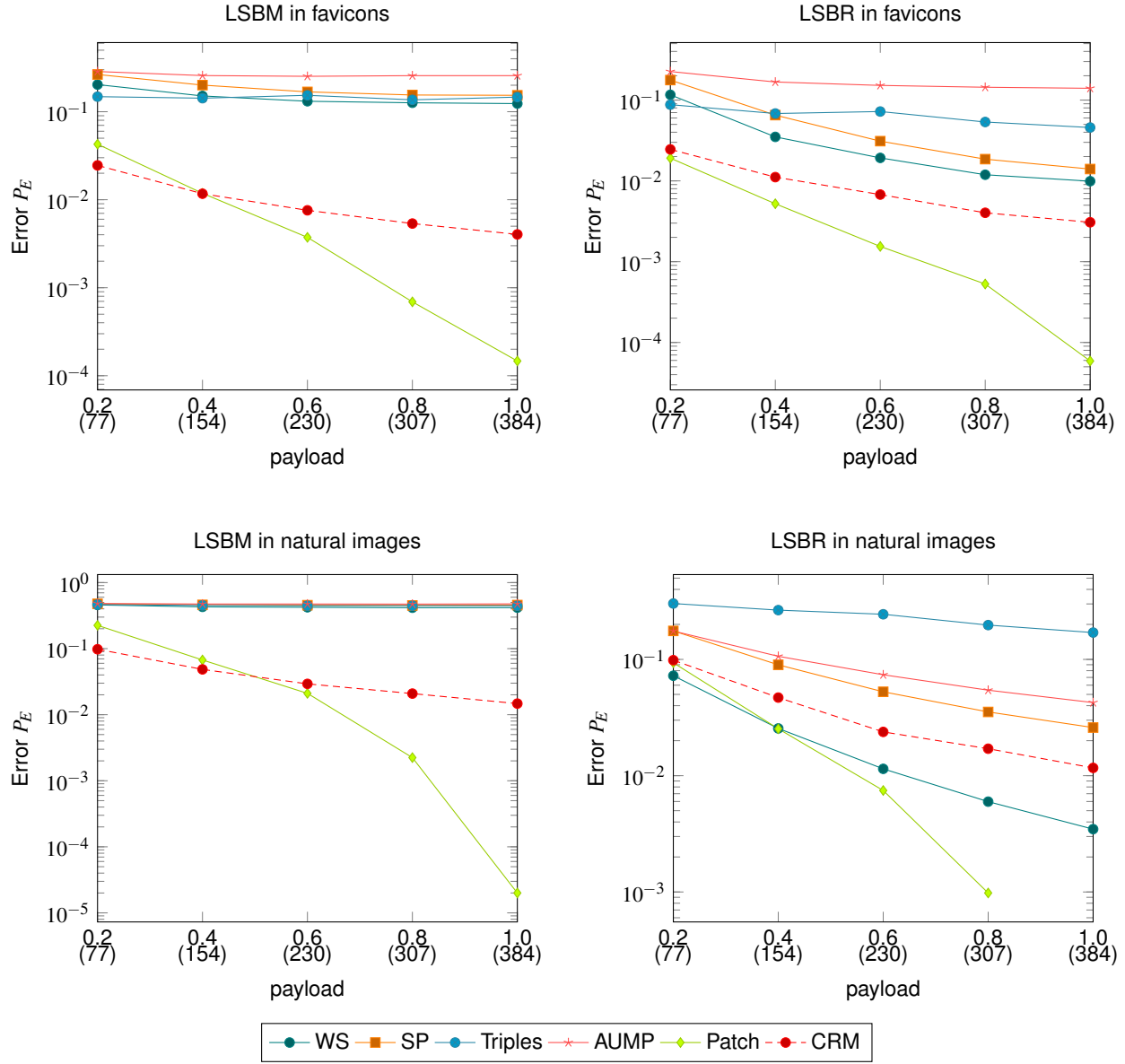


Figure 3: Equal-prior error P_E of clairvoyant detectors of least significant bit replacement (LSBR) and least significant bit matching (LSBM) in true-color favicons of size 32×32 and small natural true-color images of the same size. The error is shown with respect to relative payload measured in bits per pixel component. The number in parentheses shows the absolute length of the message in bytes. The missing point of the Patch detector on LSBR in natural images is caused by the error being zero.

used the LSBR embedding method, since the structural steganalyzers worked flawlessly.

We cannot know exactly the false positive rate of these detectors, or measure it more accurately without an even larger corpus of cover favicons. But we can gain qualitative data about “how far” the true stego icons are classified from covers by comparing their detection values with the cover distribution. Figure 4 shows the right tails of the distribution of the detectors’ outputs on cover favicons and on Vawtrak’s two stego favicons. According to this plot, Patch-based and Sample Pairs detectors have the largest gap between covers and Vawtrak’s favicons, which means that they were very certain about their decisions.

Known algorithm

Although in Vawtrak’s case the length of the hidden message is fixed, in most situations it is an unknown quantity, which can cause detectors trained on the wrong payload to malfunction; in that case the structural detectors can be superior to featured based detector. To decrease this overfitting detectors for CRM and Patch features, we trained on stego images with payload uniformly chosen from payloads $\{0.2, 0.4, 0.6, 0.8, 1.0\}$, corresponding to the solution proposed in [16]. For these experiments it is appropriate to measure the false negative rate (missed detection) at a fixed false positive rate, here 1%, which ensures that the false positive rate does not influence the accuracy measure (see [16] for a detailed explanation behind this choice).

The false negative rates of the detectors is shown in Figure 5. The results for detecting LSBR in natural images demonstrate the typical advantage of structural detectors, since Triples is the best detector (except for the smallest payload) and the difference between CRM and AUMP detectors is negligible. On the other hand structural detectors did not take advantage of the noiselessness of favicons, and naturally failed in detecting LSBM for which they have not been designed.

Conclusions

This paper studied steganography used by the Vawtrak botnet, which hides a list of Command and Control servers and other information inside favicons. The steganographic method is extremely naive, but since favicons are small and noiseless they are far from the typical images tested in the steganalysis literature. We studied experimentally how well steganalysis can detect Vawtrak’s steganography in such images. The results were positive, with many standard methods detecting Vawtrak’s favicons with a zero observed false positive rate. The fact that favicons are nearly noiseless has driven the development of new steganalytic features, called Patch.

The experimental results demonstrated that steganalysis in favicons is not the same as steganalysis in natural images: the state of the art features for color images (CRM) were outperformed by structural detectors for LSBR and by the Patch features, despite the latter being approximately 400 times faster to extract. This study shows that more work is needed on steganalysis in unusual covers, since they are used in reality. It also suggests that more work is needed for unsupervised steganalysis, since matched training data is rarely available, and universal steganalysis, since we do not know what we will face outside the laboratory.

Acknowledgements

The work of Andrew D. Ker was partially supported by Cisco Systems, Inc.

References

- [1] ars technica. A beginner’s guide to building botnets—with little assembly required. <http://arstechnica.com/security/2013/04/a-beginners-guide-to-building-botnets-with-little-assembly-required/>, 2013.
- [2] P. Bas, T. Filler, and T. Pevný. “Break Our Steganographic System”: The Ins and Outs of Organizing Boss. In *Information Hiding*, pages 59–70. Springer, 2011.
- [3] R. Cogranne, V. Sedighi, J. Fridrich, and T. Pevný. Is ensemble classifier needed for steganalysis in high-dimensional feature spaces? In *IEEE Workshop on Information Forensic and Security, Proceedings of*, 2016.
- [4] S. Dumitrescu, X. Wu, and N. Memon. On steganalysis of random LSB embedding in continuous-tone images. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages 641–644 vol.3, June 2002.
- [5] L. Fillatre. Adaptive steganalysis of least significant bit replacement in grayscale natural images. *Signal Processing, IEEE Transactions on*, 60(2):556–569, Feb 2012.
- [6] T. Filler, A. D. Ker, and J. Fridrich. The square root law of steganographic capacity for markov covers. In *IS&T/SPIE Electronic Imaging*, pages 725408–725408. International Society for Optics and Photonics, 2009.
- [7] J. Fridrich and Jan Kodovský. Rich models for steganalysis of digital images. *Information Forensics and Security, IEEE Transactions on*, 7(3):868–882, 2012.
- [8] M. Goljan, J. Fridrich, and R. Cogranne. Rich model for steganalysis of color images. In *IEEE Workshop on Information Forensic and Security, Proceedings of*, pages 185–190. IEEE, 2015.
- [9] Cisco Systems Inc. Cisco Cloud Web Security. <http://www.cisco.com/c/en/us/products/security/cloud-web-security/index.html>.
- [10] A. D. Ker. A general framework for structural steganalysis of LSB replacement. In M. Barni, J. Herrera-Joancomar, S. Katzenbeisser, and F. Perez-Gonzalez, editors, *Information Hiding*, volume 3727 of *Lecture Notes in Computer Science*, pages 296–311. Springer Berlin Heidelberg, 2005.
- [11] A. D. Ker and R. Böhme. Revisiting weighted stego-image steganalysis. In E. J. ; Wong P. W. ; Dittmann J.; Memon, N. D. Delp, editor, *SPIE on Security, Forensics, Steganography, and Watermarking of Multimedia Contents, Proceedings of*, volume 6819, page 681905, 2008.
- [12] A. D. Ker and T. Pevný. A mishmash of methods for mitigating the model mismatch mess. In *IS&T/SPIE Electronic Imaging*, pages 90280I–90280I. International Society for Optics and Photonics, 2014.
- [13] J. Křoustek. Analysis of banking trojan vawtrak. Technical report, AVG Technologies, Virus Lab, http://now.avg.com/wp-content/uploads/2015/03/avg_technologies_vawtrak_banking_trojan_report.pdf, 2015.
- [14] TREND micro. Show me the money: The monetization of KOOBFACE. <http://www.trendmicro.com/>

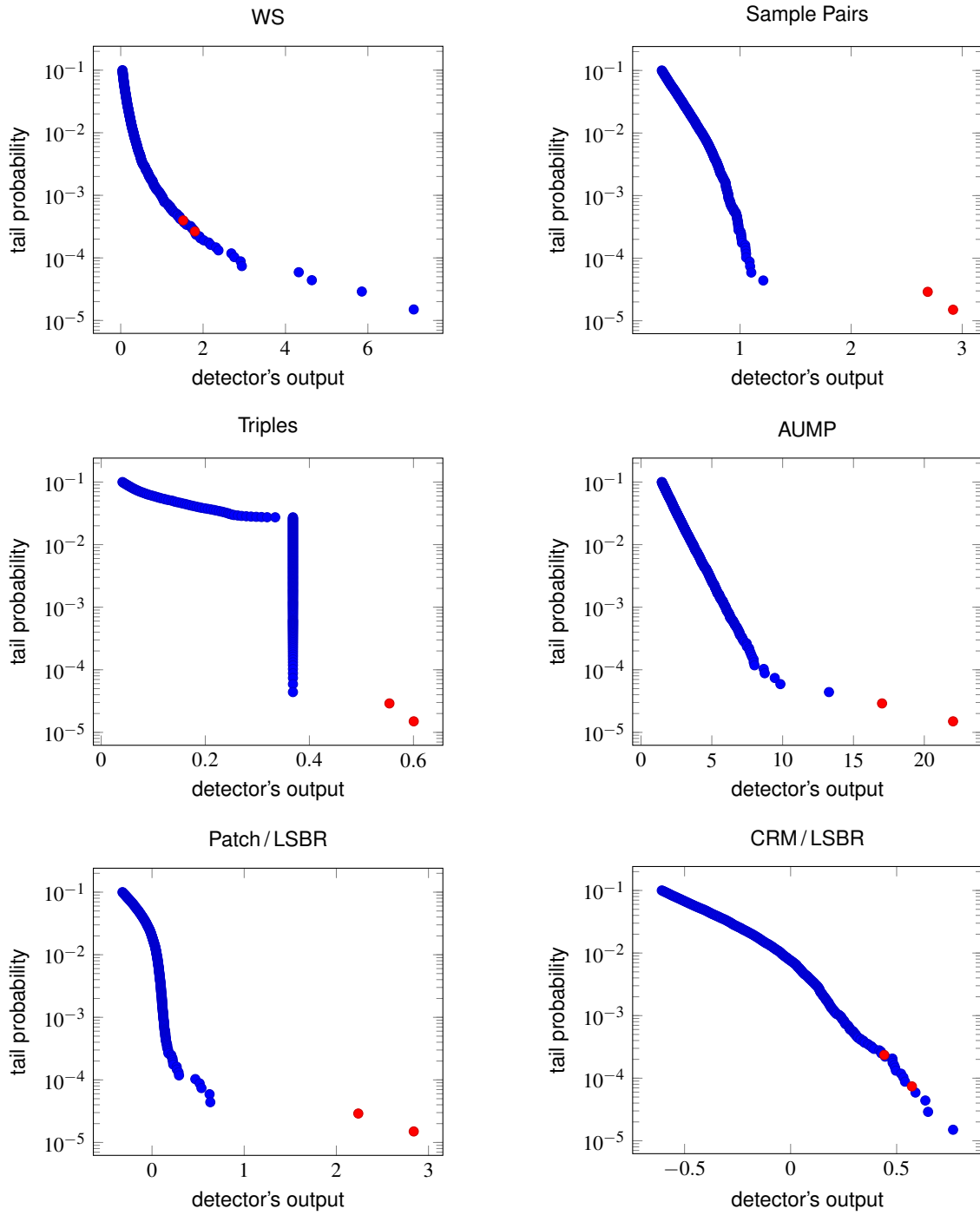


Figure 4: Tail-plots for the detectors' outputs on cover images (blue) and the two Vawtrak favicons (red). Patch and CRM detectors were trained to detect LSBR. The artefact in plot of Triples detector is caused by the output of the detector on many images being 0.3684.

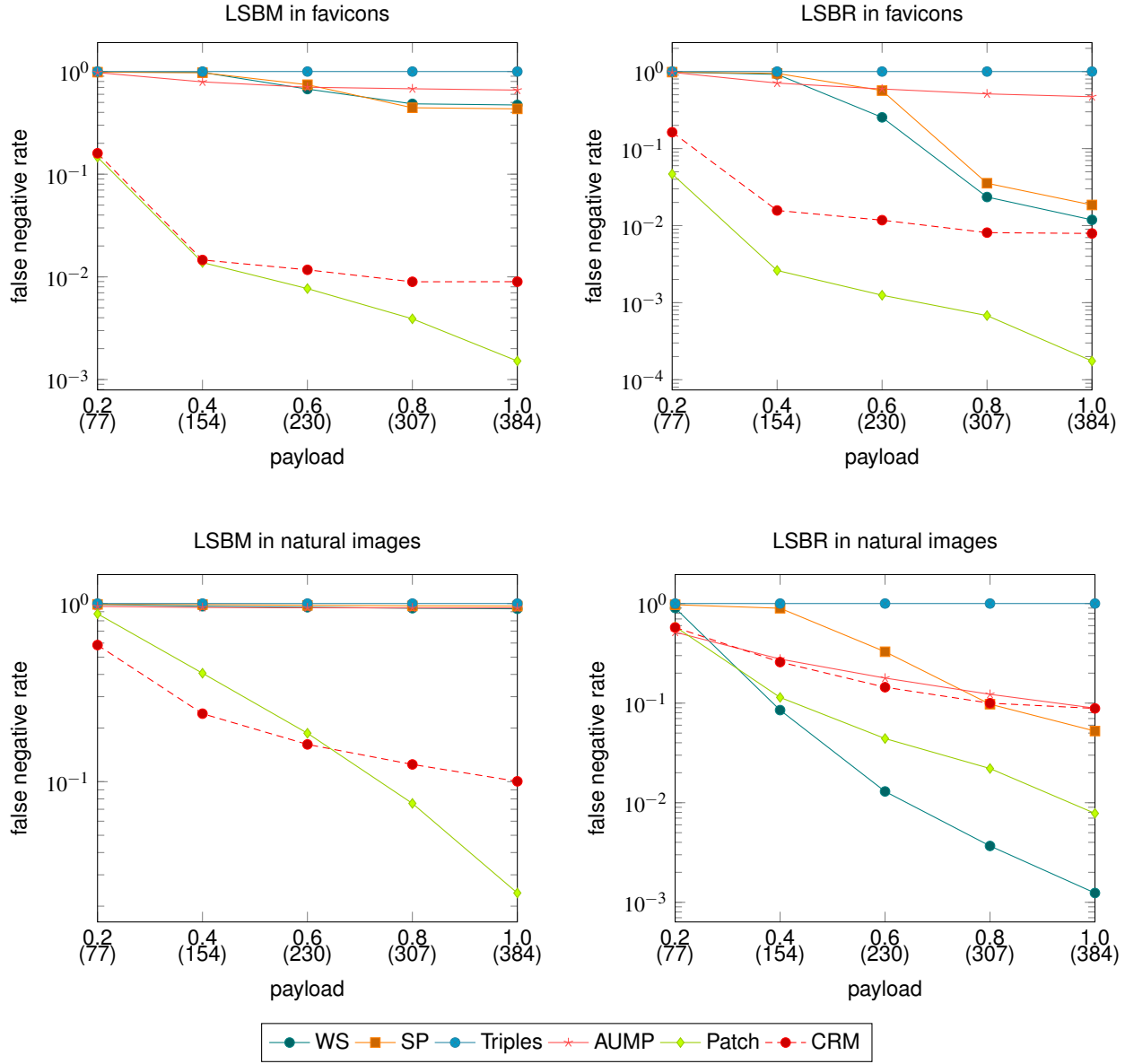


Figure 5: False negative rate (missed detection rate) of detectors knowing only the embedding algorithm of least significant bit replacement (LSBR) and least significant bit matching (LSBM) on true-color favicons of size 32×32 and small natural true-color images of the same size. The error is shown with respect to relative payload measured in bits per pixel component. The number in parentheses shows the absolute length of the message in bytes.

cloud-content/us/pdfs/security-intelligence/white-papers/wp_koobface_show-me-the-money.pdf, 2009.

- [15] TREND micro. Steganography and malware: Concealing code and C&C traffic. <http://blog.trendmicro.com/trendlabs-security-intelligence/steganography-and-malware-concealing-code-and-cc-traffic/>, 2015.
- [16] T. Pevný. Detecting messages of unknown length. *Media Watermarking, Security and Forensics XIII*, 7880:1–12, 2011.
- [17] T. Pevny and A. D. Ker. Towards dependable steganalysis. In *IS&T/SPIE Electronic Imaging*, pages 94090I–94090I. International Society for Optics and Photonics, 2015.
- [18] DARK Reading. 'xindi' online ad fraud botnet exposed. <http://www.darkreading.com/attacks-breaches/xindi-online-ad-fraud-botnet-exposed/d/d-id/1323212>, 2015.
- [19] R. L. Rivest and J. C. N. Schuldt. Spritz—a spongy RC4-like stream cipher and hash function. <http://people.csail.mit.edu/rivest/pubs/RS14.pdf>, 2014. Presented at Charles River Crypto Day (2014-10-24).
- [20] B. Stone-Gross. Malware analysis of the lurk downloader. Technical report, Dell SecureWorks Counter Threat Unit, <http://www.secureworks.com/cyber-threat-intelligence/threats/malware-analysis-of-the-lurk-downloader/>, 2014.
- [21] FBI Testimony. Taking down botnets. <https://www.fbi.gov/news/testimony/taking-down-botnets>, 2014.
- [22] Tripwire. Hackers exfiltrating data with video steganography via cloud video services. <http://www.tripwire.com/state-of-security/incident-detection/hackers-exfiltrating-data-with-video-steganography-via-cloud-video-services/>, 2014.