A Picture is Worth a Thousand Words, Literally: Deep Neural Networks for Social Stego

Philip Tully and Michael T. Raggo

Abstract and Introduction

Images, videos and other digital media provide a convenient and expressive way to communicate through social networks. But such broadcastable and information-rich content provides ample illicit opportunity as well. Web-prevalent image files like JPEGs can be disguised with foreign data since they're perceivably robust to minor pixel and metadata alterations. Slipping a covert message into one of the billions of daily posted images may be possible, but to what extent can steganography be systematically automated and scaled?

To explore this, we first report the distorting side effects rendered upon images uploaded to popular social network servers, e.g. compression, resizing, format conversion, and metadata stripping. Then, we build a convolutional neural network that learns to reverse engineer these transformations by optimizing hidden data throughput capacity. Pre-uploaded and downloaded image files teach the network to locate candidate pixels that are least modifiable during transit, allowing stored hidden payloads to be reliably recalled from newly presented images. Deep learning typically requires tons of training data to avoid overfitting. But data acquisition is trivial using social networks' free image hosting services, which feature bulk uploads and downloads of thousands of images at a time per album.

We show that hidden data can be predictably transmitted through social network images with high fidelity. Our results demonstrate that AI can hide data in plain sight, at large-scale, beyond human visual discernment, and despite third-party manipulation. Steganalysis and other defensive forensic countermeasures are notoriously difficult, and our exfiltration techniques highlight the growing threat posed by automated, AI-powered red teaming.

Contents

The Evolution of Covert Communications

Steganography has been practiced for millennia. Ancient Chinese dynastic cultures hid military secrets as early as 525 BC by covering message-imbued silk in wax and rolling it into a ball. German spies used photographically produced microdots to steal uranium design information, production statistics, and building schematics during World War II. The first documented digital steganography occurred in 1985 when employees of a small company communicated over restrictive channels using their newly minted personal computers.

Whether it's intended for tactical battlefield advantage, international espionage, or advanced persistent cyber attacks, steganography transcends due to its fundamentally covert design. Its goals remain the same across technologies and use cases. As the processing power, network bandwidth, storage capability, file format heterogeneity and mobility of computing devices continue to grow, steganography will live on and continue to frustrate forensic investigators [1].

These days, social networks make for particularly attractive steganography conduits because they:

- provide public access to massive, prioritizable lists of targets and recipients,
- are highly trafficked, making it hard to distinguish malicious signal from noise,
- include convenient broadcasting syntax mechanisms like #hashtags,
- feature both manual and programmatic search capability,
- possess an undeserved reputation for trust and safety,
- lack the precedent for security-mindful engagement that older channels like email enjoy,
- exist outside traditional perimeter and endpoint security,
- are culturally ingrained, meaning blocking employee access is often unviable,
- provide free image hosting services, and
- allow users to resize or crop uploaded photos using backend software like ImageMagick, which makes web servers vulnerable to malicious steganographic code execution (i.e. ImageTragick, CVE-2016-3714) [2].

What about stego-based social media attacks taking place in the wild? Evidence abounds. Encrypted callback URLs posted to Twitter can connect to C&C servers and install malicious GIF-embedded payloads [3]. HAMMERTOSS scanned tweets for hashtags and images, which were subsequently decoded to execute C&C instructions [4]. A malicious plugin can similarly search Google+ for PNG files with encrypted C&C configurations [5]. CryLocker ransomware compiled victims' information into fake PNGs and uploaded them to Imgur, which broadcasted them to C&C IP addresses to alert operators of fresh infections [6].

Whitehat-inspired social stego research has also been carried out. SneakyCreeper demonstrated that Twitter, Tumblr, and SoundCloud were susceptible to base64-encoding messages into audio and image files [7], a proof of concept C&C trojan used a steganographic decoder to execute code hidden within Instagram images [8], and MP3 ID3v2 metadata

manipulation was shown to circumvent media sharing services [9]. The techniques presented in this work seek to generalize some of these previous approaches, and in the same vein, raise awareness about steganography-based attacks on social networks.

DIY Data Exfiltration

Despite the aforementioned risks, there is a general belief that steganography doesn't justify defense. We analyze the nuances of uploaded image distortion across popular social networks in order to enumerate the level of sophistication being incorporated against steganography-based attacks.

One method is to use the designated landmarks of an image file, like its metadata and end-of-file (EOF) markers, to deposit the payload. Metadata fields such as Exchangeable image file (Exif), International Press Telecommunications Council (IPTC) and Extensible Metadata Platform (XMP) formatted sectioned off text areas are usually reserved for technical information and photo capture statistics. The EOF marker occurs after the last byte of the file and indicates to image reading programs how much storage space to allocate in advance. Hidden data can be appended beyond the EOF marker, since many programs may ignore anything placed afterwards.

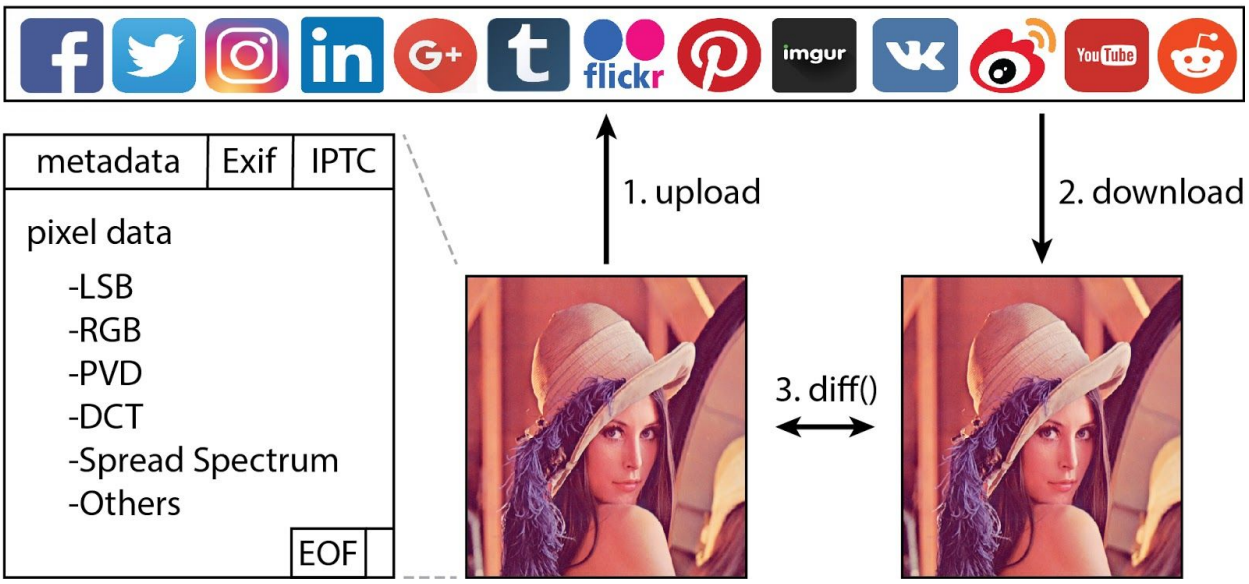

Fig 1: Schematic of the workflow carried out during exploratory analysis. A Lenna cover image file is injected with foreign data according to the techniques outlined by its corresponding bytemap on the left. It is then 1. uploaded to a social network, 2. subsequently downloaded, and 3. programmatically compared to the pre-uploaded version to ascertain what operations are performed on the image by each social network platform.

In terms of image pixel data, digital steganography techniques can be categorized into two separate domains: spatial and frequency. Spatial domain steganography involves directly manipulating pixel intensities, whereas frequency domain steganography manipulates the orthogonal transform of the image. Generally, the spatial domain is less robust but has a higher storage capacity than the frequency domain.

We conduct an exploratory analysis by uploading images with Least Significant Bit (LSB), Red Green Blue-based (RGB), and Pixel Value Difference (PVD) spatial domain steganography, and Discrete Cosine Transform (DCT) and Spread Spectrum frequency domain steganography. We test each of these techniques across 14 different social networks: Facebook, Twitter, Instagram, Pinterest, LinkedIn, Google+, Tumblr, Flickr, Imgur, VK, Sina Weibo, YouTube, Slack, and Reddit.

Each network can have different fields suitable for image upload, for example on Facebook the profile picture versus the cover photo. For all of 14 networks, we test every uploadable field across all supported image types (eg. JPEG, PNG, BMP, GIF, etc). We also test images with different dimensions and file sizes, and try reuploading previously uploaded images to test to see whether social networks leave recognizable traces that would allow images to bypass certain upload processing steps in the future. We then present a chart of commonalities and variations in steganographic efficacy measured across the comprehensive social landscape.

We report the distortions that each social network renders on images uploaded to their servers, including image resizing, format conversion, metadata stripping, and compression. Compression is often leveraged in order to dynamically render large images at smaller sizes to keep bandwidth-throttled social network users happy. It can be either lossy or lossless, where lossy compression increases the likelihood of payload clobbering because excess image data is not preserved.

A Chrome extension called SecretBook [10] managed to minimize payload clobbering when images were uploaded to Facebook by reverse engineering the social network's lossy compression algorithm. It then applied similar processing steps to pre-compressed images prior to Facebook upload and demonstrated robust social steganography for payloads up to 140 characters in length. For social networks that were found to compress and distort uploaded images, we sought to develop a generalizable, data-driven way of similarly performing steganography.

Deep Neural Networks for Social Stego

The rise of data emitted by mobile phones, social networks, IoT devices, cameras and software logs has inspired new AI applications to information security. Such available data, together with open source deep learning software libraries and cloud computing resources have conspired to make AI more accessible. ConvNets are powerful algorithms that've been shown to generalize

across many computer vision tasks like object classification, facial recognition and video analysis. They comprise multiple feedforward layers of neurons that nonlinearly extract features from input data. Each convolutional layer has an associated parameter set and can learn filters that activate when certain spatial locations are detected within their inputs. Its weights are optimized through back propagation.

We build a ConvNet that learns to reverse engineer image upload manipulations by locating optimal pixel coordinates for embedding a hidden payload. We pose the problem as a regression task that uses a generic ConvNet architecture with several stacked layers of convolutions fed through rectified linear units (ReLUs) and a final smooth L1 loss regression layer. The input to the ConvNet is the set of pixels from pre-uploaded images, and the output of the final layer is the set of candidate pixels that are the least-likely-to-be-clobbered while transiting through the social network. In other words, the output of the ConvNet represents spatial locations that conceal the largest hidden payloads with minimal deformation.

Deep neural networks require massive amounts of training data in order to avoid overfitting. Luckily, in competition to expand their user bases, social networks typically provide free and user-friendly functionality to bulk upload and download images off the shelf. The more content shared, the greater the opportunity to profit. Furthermore, their image upload APIs tend to be permissive in order to incentivize developers to build apps that post photos or manage photos and albums. In many cases, this simplifies the problem of acquiring training data even further by enabling programmatic uploads and downloads.

When fed tens of thousands of training images, the trained ConvNet can predict with high fidelity which pixel locations are ideal for storing hidden payloads. We demonstrate how learned locations typically correspond to the more complex and “busier” regions of an image. For example, in a photo with horses galloping in a field, horses are better payload location targets than the more uniform grass and sky in the background. The ConvNet is built with Google’s TensorFlow, which is an open source software library that conveniently exposes all functions and classes necessary for our purposes through it’s API.

To evaluate the ConvNet, we consider two separate metrics: minimal visual dissimilarity between the pre-uploaded and downloaded image in terms of peak signal to noise ratio, and maximal payload capacity in terms of byte survivability. We show that our techniques allow for more robust and less detectable transmission of payloads. We attribute our results to the unique risks associated with social media and our ability to leverage statistical approaches on huge amounts of image data.

Conclusion

Steganography is imperceptible to the naked eye by design. Our approach is robust to aftereffects like image filters, since this information is directly available from the downloaded

images during training. However, graceful degradation can make it difficult to store syntactically rigid payloads like source code or malware. In these cases, single-character changes can break code compilation or obfuscate fault intolerant payloads. This is especially the case for longer payloads because recovery rates worsen as the size of the hidden data increases. Implementing error-correcting codes, duplicating the payload across images or fragmenting the payload across single or separate images could help address these shortcomings.

This work joins a short but growing list of offensive techniques that automate a traditionally manual attack workflow using AI, including micro-targeted social engineering [11], password cracking [12] and captcha subversion [13]. Offensive AI is easier to implement than its defensive counterparts; it can be trained using either unsupervised learning algorithms or supervised ones with cheap-to-label data. This labeling bottleneck will create headaches for blue teamers, who will struggle to keep with the extra money, time and effort taken to generate reliably instructive data samples. Success rates are far more important for the blue team than they are for the red team because of what's at stake, too. Accelerating AI accessibility will only magnify this problem. With more frequent open-source initiatives and cheapening access to powerful cloud-based computing resources like GPUs, the barrier to entry for applied AI will continue to retreat.

Not all hope is lost though. After all, the best defense is a good offense. Adversarial learning and data-driven security are poised to transform modern cyber defenses. The rise of machine hacking will harden industry security by plugging up previously unknown holes, and the sooner this is realized, the better.

Our approach excels because user-hungry social networks rely on freemium and ad-based business models, which provide cloud-based services like image hosting off the shelf. It can be extended by incorporating more advanced cryptography to turn the steganographic payload into a cipher for better camouflage. Social media backdoors can be used to exfiltrate sensitive data from within private networks, perform reconnaissance, distribute malware or maintain contact with C&C infrastructures [3-6]. This work has implications for insider threats, terrorism, copyright infringement, and corporate or nation state espionage. Finally, it underscores the problem of social media data loss prevention. We share this automated steganography enabling tool in order to raise awareness about cloaked data within public information streams, and more generally to raise awareness about security risks associated with social networks.

References

[1] "Data Hiding: Exposing Concealed Data in Multimedia, Operating Systems, Mobile Devices and Network Protocols", Mike Raggo and Chet Hosmer, 2012.

[2] "ImageTragick", @stewie and Nikolay Ermishkin, May, 2016.

[3] “A Closer Look at MiniDuke”, Marius Tivadar, Bíró Balázs and Cristian Istrate, BitDefender, February, 2013.

[4] “HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group”, FireEye, July 2015.

[5] “BE2 extraordinary plugins, Siemens targeting, dev fails”, Kurt Baumgartner and Maria Garnaeva, Kaspersky Lab, February, 2015.

[6] “CryLocker”, Malware Hunter Team, September, 2016.

[7] "Getting the data out using social media", Dakota Nelson, Gabriel Butterick, Byron Wasti and Bonnie Ishiguro, *BSides Las Vegas,* 2015.

[8] “Instegogram: Exploiting Instagram for C2 via Image Steganography”, Amanda Rousseau, Hyrum Anderson and Daniel Grant, *DEF CON 24 Village Talks*, August, 2016.

[9] “What’s lurking in MP3s that can hurt you?”, Chet Hosmer and Mike Raggo, *DEF CON 24 Skytalks*, August, 2016.

[10] “Secretbook”, Owen-Campbell Moore.

[11] “Weaponizing Data Science for Social Engineering: Automated E2E Spear Phishing on Twitter”, John Seymour and Philip Tully, *Black Hat USA* 2016.

[12] "Fast, Lean and Accurate: Modeling Password Guessability Using Neural Networks," William Melicher, Blase Ur, Sean M. Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin and Lorrie Faith Cranor, *Proceedings of USENIX Security*, 2016.

[13] "I am robot:(deep) learning to break semantic image captchas, " Suphannee Sivakorn, Iasonas Polakis and Angelos D. Keromytis. *2016 IEEE European Symposium on Security and Privacy*.