Editors: D. Balzarotti, davide.balzarotti@eurecom.fr | W. Enck, whenk@ncsu.edu | T. Holz, thorsten.holz@rub.de | A. Stavrou, astavrou@gmu.edu

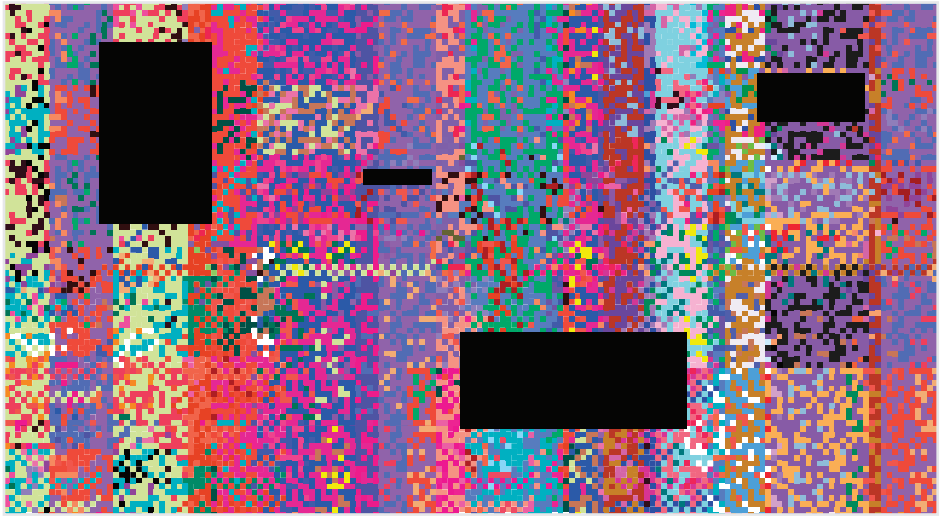
Never Mind the Malware, Here's the Stegomalware

Luca Caviglione | National Research Council of Italy
Wojciech Mazurczyk | Warsaw University of Technology and FernUniversität in Hagen

In recent years, the use of steganographic techniques has been progressively observed to endow threats with cloaking capabilities and launch sophisticated attack campaigns. This requires partially rethinking the perception of malware.

Steganography is an umbrella term for a set of techniques allowing to hide information in plain sight. There is a common belief that steganography mainly belongs to academic books, and its usage is almost confined to research papers. This is especially true when moving to computer science: apart from successful ideas that enforce copyrights and detect alterations of digital media[1] or track traffic flows,[2] several applications only seem like articulated mental exercises. However, amid tremendous growth in the diffusion of malware that started in 2014, many security researchers and experts noticed that attacks were becoming harder to detect, and many threats wandered undisturbed on the Internet for years. As an example, Regin was discovered in 2014, but its original inception can be dated back to 2003.

Even if such an increased stealthiness is the result of a mix of coding techniques (e.g., binary obfuscation), defensive approaches (e.g., antiforensics routines and memory encryption), and architectural optimizations (e.g., multistage loading to postpone the retrieval of malicious payloads), researchers found that steganography was an essential ingredient of this success.[3] General recognition of the diffusion of malicious software endowed with steganographic capabilities is supported by the increasing popularity of the term *stegomalware*.



©SHUTTERSTOCK.COM/LOCAL_DOCTOR

Actually, such an expression has not been coined to describe the new wave of threats that began to emerge in 2014. Instead, it has been recently borrowed from a past work dealing with malware targeting Android that exploited steganography to conceal malicious code within the resources of applications.[4]

"Le Menu"

As mentioned before, the ultimate goal of steganography is to make information invisible. This is the primary difference when compared to classical obfuscation techniques, which aim at rendering the information difficult to comprehend, especially to prevent reverse engineering or distillation of signatures. For instance, binary obfuscation exploits a set of manipulations (e.g., adding nop/junk code, reordering instructions, nesting indirections, and renaming packages) to make the available code difficult to understand or to match against detection rules. To give a practical example, a standard binary obfuscation technique considers reassigning registers: for the case of Win/Intel Regswap malware, registers EAX, EBX, and EDX are reassigned to the different sequence EBX, EDX, and

Digital Object Identifier 10.1109/MSEC.2022.3178205
Date of current version: 6 September 2022

EAX.[5] Instead, when using steganography, detection is eluded by converting the sequence of instructions into an intermediate form (e.g., ASCII or Base64), which is then hidden by manipulating pixels of an image or cloaking it in metadata.[3] Similarly, for the case of network conversations, traffic obfuscation creates a sequence of packets impersonating another protocol or exhibiting standard features (e.g., size of the packet or used ports), which are considered safe. On the contrary, stegomalware primarily exploits network covert channels, which are created by hiding malicious data in legitimate traffic to enable cloaked communication paths. As a possible example, the attacker could activate a botnet by hiding commands in the interpacket delays of a target stream.

Therefore, the common definition of stegomalware as *a malware using some form of steganography to remain undetected* should be considered too reductive. In fact, recent samples observed "in the wild" take advantage of numerous cloaking techniques that also extend to the broader domain of information hiding. In summary, stegomalware often comprehends mechanisms to

- hide malicious code or other resources (e.g., libraries, scripts, and configuration files) within innocent-looking data to avoid detection, blockages, or content-filtering disciplines
- drop payloads on the host of the victim or retrieve additional executables to reduce the footprint of the threat, elude antivirus software, or make the reconstruction of the attack more difficult
- implement covert channels to exfiltrate information and exchange commands as well as to circumvent traffic policies or intrusion-detection systems
- allow processes to exchange data even if confined in separate execution enclaves or hardware entities (e.g., sandboxes, containers, or CPU cores).

As detailed later, the utilization of steganography for attack purposes observed during the years mutated and is not distributed in an even manner.

Name Names

In May 2021, a threat able to conceal information in the metadata of profile pictures of the Steam online gaming platform had been isolated. Similar to other attacks, the manipulated profile picture cannot launch executable code: instead, it only serves as the hidden carrier for another malicious stage. Upon execution, the threat checks for the presence of Microsoft Teams and evaluates the reachability of Twitter to implement command and control (C&C) operations or orchestrate bots.

The ultimate goal of steganography is to make information invisible.

In July 2020, a variant of the OilRig malware was discovered. Compared to previous incarnations, the most interesting difference concerns the use of steganography to implement a C&C channel nested in email traffic, i.e., data are hidden in innocent-looking pictures exchanged as attachments. Steganography was also a key factor in a large-scale attack launched in May–June 2020 against industrial institutions of Japan, Italy, the United Kingdom, and Germany. In more detail, Trojan.MSExcel.Agent.be and Trojan.PowerShell.Generic were used to infect the victims and retrieve additional payloads, including configuration files. To avoid detection and reduce their footprint, trojans downloaded an image from a public service like imgur.com, which contains hidden malicious code. In the same period, another malware taking advantage of steganography and spreading via USB mass-storage devices was identified. Named *USBFerry*, it exploits the BKDR_IDSHELL.ZTFC-A backdoor, which injects secret data both in Domain Name System (DNS) traffic and JPG images while communicating with a remote controller. A similar approach was also observed in Titanium earlier in 2019: in this case, data were hidden in PNG files.

The growing interest in endowing malicious software with steganographic techniques is also demonstrated by the waves of attacks that periodically interested several online stores from late 2020 until the end of 2021. With the blanket term *Magecart*, security experts started to denote several cybercriminal groups targeting e-commerce platforms to steal credit card numbers and sell them for profit. As an example, in the first quarter of 2020, cybercriminals compromised the Tupperware website by cloaking a web skimmer within a PNG file looking absolutely coherent with the content displayed in the browser of the victim. The spectrum of data-hiding techniques at the basis of the attack is quite broad. For instance, malicious code is concealed in comment chunks of PHP and CSS files of the e-commerce site, whereas data of the victim are embedded in JPG images via malicious code injected into the checkout page.

A wide array of attacks hiding contents in images was isolated in 2019. For instance, LokiBot concealed

malicious .zipx files within a PNG to improve its resistance against reversing and forensics attempts. Multistageloadingmechanismsleveraging steganographically-modified images were also observed in the StarCruft threat. Nevertheless, an updated version with steganographic capabilities of the Cardinal Remote Access Trojan threat (first appearing in 2017) was observed in 2019 while targeting FinTech organizations.

For the sake of brevity, we limited our discussion to the most recent threats isolated by security experts and researchers. A list of the most notable and known stegomalware observed in the wild is summarized in Table 1. As shown, the table reports where information is hidden, the cloaking techniques used, and for what purpose. A more comprehensive and updated list is available online.[6]

The Case of Invoke-PSImage

The Invoke-PSImage[7] is a good synecdoche of the great difficulties in speculating when steganography will be used. In essence, Invoke-PSImage allows embedding a PowerShell script in pixels of an image by altering the four least-significant bits of the related green and blue channels. Even if simple and with several limitations, the method has been published on GitHub and rapidly spread in security-related forums. Surprisingly, Invoke-PSImage was used in the first attack campaign against the Pyeongchang Olympic Games just two days after its release.[8] This proves that the availability of "running code" is one of the most important criteria for the success of an offensive technique. In recent years, Invoke-PSImage has been exploited by Powload (spotted in the first half of 2018), and it is still in the toolbox of different incarnations of Emotet and Bebloh.[9,10] It is also the basis of stages for dropping payloads on the host of the victim in malicious software like the Greystar ransomware and some variants of Ursnif.[11] Another notable utilization concerns the creation of backdoors, as it happens in Bandook.[12]

What's Next?

It is pretty clear that the broad set of techniques and contents used by stegomalware leads to an asymmetry between the attacker and defender. Moreover, if the cybercriminal suspects that a carrier will be monitored, he/she can simply move to another one, making the design of one-size-fits-all countermeasures almost impossible. Besides, many security professionals are curious rather than concerned about the risks arising from the use of steganography. Thus, the availability of both open source and commercial products (e.g., firewalls and antivirus) is highly fragmented or specialized. For instance, many intrusion detection systems are not designed to handle network covert channels out of the box and require development of specific rules.[13] For the case of threats targeting images, some ad-hoc products are becoming available, e.g., the Steganography Defensive Initiative of McAfee,[14] but they do not appear mature enough for large-scale utilization.[10] The researchers and security experts genuinely concerned with stegomalware should then consider the following four principles when engineering defensive strategies.

1. *Opportunity makes the thief*: Steganographic and hiding methods mainly take advantage of unneeded information, imperfect isolation, or ambiguities that can be manipulated to store secrets. Therefore, future hardware, application programming interfaces, multimedia files, and network protocols should enforce some steganographic-resistance-by-design criteria. This may require the definition of suitable formal methods or testing practices as well as the design of "fragile" resources that stop working when altered, even if minimally. For instance, case-insensitive schemes should be avoided to prevent the encoding of secrets in keywords or tags, e.g., <HEAD> versus <head>, and padding patterns or values of unused bits should not be discretional. Violations should render the service unusable or at least raise a warning.
2. *Decouple*: Detecting and mitigating stegomalware requires inspecting multiple artifacts at once, which can be very different (e.g., binaries, metadata, and network packets). Even if each threat could have its own traits demanding for specific signatures (e.g., volumes of syscalls or patterns in response bodies of HTTP traffic), the technology used to gain visibility over hardware and software may be unique. An approach to mitigate the burden needed to chase attackers hiding data in "unexpected" places could benefit from a decoupled design. Countermeasures

Invoke-PSImage allows embedding a PowerShell script in pixels of an image by altering the four least-significant bits of the related green and blue channels.

Table 1. The most recent and notable stegomalware observed in real-world attacks.

Name of the Attack	Description/Goals	Type	Used Techniques
Steamhide	Downloads malware via a profile image published on the Steam platform.	Image	Malicious encrypted code is placed in the PropertyTagICCProfile.
OilRig	Targeted Middle Eastern telecommunication organizations. It enables a covert communication using emails and image steganography.	Image	Secrets are embedded in BMP images and sent as an attachment of fake emails.
Trojan.PowerShell.Generic	Aims enterprises in Japan and Europe. A malicious XLS file containing the Trojan.MSExcel.Agent.be macro is executed to retrieve an image hiding the Trojan-PSW.PowerShell.Mimikatz.	Image	The PowerShell script randomly selects an URL from an embedded list and gets an image from imgur.com or imgbox.com. Steganography is used to embed/extract a malicious script from the downloaded image.
IcedID / BokBot	Banking trojan that mainly aimed at U.S. bank customers but also at telecommunications (AT&T) and mobile communications (T-Mobile) companies. The first- and second-stage downloaders transmit an image from the C&C server.	Image	The first-stage loader utilizes an image to extract the second-stage loader. Then, the latter retrieves the shellcode, the IcedID core, and various configuration files.
USBferry	It targets Taiwanese and Filipino physically-isolated networks used for military purposes. It masks backdoor (BKDR_IDSHELL.ZTFC-A) routines to evade antimalware and network perimeter detection.	Image	The malicious code is hidden in JPG or PNG files.
Magecart	Threat that is implementing a credit card skimmer. It steals credit card information from e-commerce platforms	Image	Credit card data or malicious JavaScripts are appended at the end of the structure of the JPG/PNG file.
Powload	One of the most pervasive threats in the North American region during 2018. It was used to deliver information-stealing payloads such as Emotet, Bebloh, and Ursnif.	Image	It utilized the publicly-available steganographic tool Invoke-PSImage to embed malicious PowerShell scripts into innocent-looking PNG files.
LokiBot	To resist reversing and forensics attempts, it conceals malicious files within a digital image.	Image	The first variant hides malicious ZIPX file attachments inside a PNG image file. The second embeds encrypted binary in a JPG. The third appends an encrypted DLL to a BMP file.
Okrum/Ketrican	Targeted diplomatic missions in Slovakia, Belgium, Chile, Guatemala, and Brazil. It downloads malicious code to evade antimalware and network perimeter detection.	Image	Stage-1 loader containing the backdoor is embedded in a valid PNG file.
Cardinal RAT	Group of attacks targeting Israeli-based FinTech companies that develop software related to foreign exchange and cryptocurrency trading. The malware hides itself using .NET and BMP files.	Image	After execution, the malware reads the BMP file, parses pixel data from the image, and decrypts the result using a single-byte XOR key.
USBferry	It targets Taiwanese and Filipino physically-isolated networks used for military purposes. It exploits network traffic for two purposes. In the first case, DNS traffic is used to communicate with the C&C server and provide information on the infected host. In the second case, a portion of HTTP request is used to create a web shell to let the attacker remotely control the infected host.	Network traffic	Communications with the backdoor controller are cloaked in various DNS messages, whereas commands for controlling the infected host are hidden in the cookie header of HTTP traffic.

(*Continued*)

Table 1. The most recent and notable stegomalware observed in real-world attacks. (*Continued*)

Name of the Attack	Description/Goals	Type	Used Techniques
DarkHydrus	It was used to carry out a credential-harvesting attack on an educational institution in the Middle East. DNS messages are used to cloak C&C communication.	Network traffic	DNS tunneling is applied to enable communication with C&C server. A variety of different DNS query types is utilized for this purpose.
Okrum/Ketrican	It targets diplomatic missions in Slovakia, Belgium, Chile, Guatemala, and Brazil. A part of HTTP request is used to cloak the communication between the C&C server and the infected host.	Network traffic	C&C messages are concealed in HTTP traffic, i.e., in Set-Cookie and Cookie headers of HTTP requests.
OilRig	It targets Middle Eastern telecommunication organization. It utilizes DNS traffic to communicate with the C&C server.	Network traffic	The threat uses DNS tunneling implemented by hiding its C&C communications within TXT records.
IcedID/BokBot	Family of banking trojan that mainly aimed at U.S. bank customers but also at telecommunications (AT&T) and mobile communications (T-Mobile) companies. A part of HTTP request is used to transmit various information related to the infected system to the C&C server.	Network traffic	The system's fingerprinting data are hidden in the Cookie header of the HTTP traffic.
Magecart	Threat that is implementing a credit card skimmer. It steals credit card information from e-commerce platforms.	Text	It conceals malicious code using comments in PHP or CSS files.
Astaroth	This attack campaign exclusively targeted Brazil and featured lures explicitly designed to tailor to Brazilian citizens. It exploited popular services like YouTube.	Text	The description of YouTube channels hides the encoded and encrypted URLs of C&C servers.
Platinum APT	It targeted diplomatic, government, and military entities in South and Southeast Asian countries. It exploited HTML for concealing C&C communication.	Text	C&C messages are hidden in the order of HTML attributes, and the encryption key used to decode the secret content was encoded in the sequence of spaces among HTML tags.

against malware endowed with cloaking capabilities should be composed of specific threat-dependent functionalities and general inspection technologies, like the extended Berkeley Packet Filter.[15,16]

3. *You will never know*: A part of the functionalities of the Duqu malware was implemented in an encrypted dynamic-link library (DLL) hidden in a JPG file. Yet, its presence had been witnessed by temporary files named *~DQ*. This short anecdote is to say that despite being cloaked, stegomalware may leave indications of its presence in unexpected places. Hidden data may inflate the size of a file, parasitic usages of network traffic may alter the distribution of bits in the header of a packet, and image steganography may produce traits in the RGB histogram. Indeed, where the stegomalware will impact cannot be foreseen, and relationships between signatures and cloaked data often cannot be captured by common sense. To this aim, artificial intelligence allows for partially taming the blind nature of the defender against information-hiding-capable threats.[17]
4. *Follow the code*: The message of the Invoke-PSImage parable should be taken in high regard, especially with the increasing diffusion of the crime-as-a-service model allowing to engineer and implement a threat by using third-party code and facilities. In fact, if an adequate and publicly available method will become part of a malicious kit or at the basis of a threat progeny, its adoption turns out to be quite impossible to tame. At the same time, scouting the Internet for ideas, tools, and proof-of-concept implementations exploiting steganography could be a worthy effort in the perspective of anticipating cloaking strategies or chasing alicious threats before they disappear.

Enriching malware with information hiding capabilities is a trend that is expected to grow. In fact, even simple techniques have been demonstrated to be effective against many security tools and routine practices. Thus, stegomalware is here to stay, and security experts should intensify research activities, especially in the design of efficient countermeasures. ■

Acknowledgment

This work was supported by the Horizon 2020 Program through the Secure Intelligent Methods for Advanced RecoGnition of Malware and Stegomalware Project under agreement number 833042.

References

1. V. M. Potdar, S. Han, and E. Chang, "A survey of digital image watermarking techniques," in *Proc. 3rd Int. Conf. Ind. Inform.*, 2005, pp. 709–716.
2. A. Iacovazzi and Y. Elovici, "Network flow watermarking: A survey," *IEEE Commun. Surv. Tuts.*, vol. 19, no. 1, pp. 512–530, 2017, doi: 10.1109/COMST.2016.2604405.
3. W. Mazurczyk and L. Caviglione, "Information hiding as a challenge for malware detection," *IEEE Secur. Privacy*, vol. 13, no. 2, pp. 89–93, Mar./Apr. 2015, doi: 10.1109/MSP.2015.33.
4. G. Suarez-Tangil, J. E. Tapiador, and P. Peris-Lopez, "Stegomalware: Playing hide and seek with malicious components in smartphone apps," in *Proc. 10th Int. Conf. Inf. Security Cryptol.*, 2014, vol. 8957, pp. 496–515.
5. I. You and K. Yim, "Malware obfuscation techniques: A brief survey," in *Proc. Int. Conf. Broadband, Wireless Comput., Commun. Appl.*, Fukuoka, Japan, 2010, pp. 297–300.
6. "Steg in the wild," GitHub, San Francisco, CA, USA. Accessed: Jun. 2022. https://github.com/lucacav/steg-in-the-wild
7. "Invoke-PSImage," GitHub, San Francisco, CA, USA. https://github.com/peewpw/Invoke-PSImage
8. "Cyber security briefing: A monthly recap of technology and information risk," Pinkerton, Feb. 2018. Accessed: Jun. 2022. https://pinkerton.com/media/our-insights/briefings/sources/cybersecurity-newsletter-2-18.pdf
9. "Internet security threat report," Symantec, Feb. 2019. Accessed: Jun. 9, 2022. https://docs.broadcom.com/doc/istr-24-2019-en
10. R. Han, C. Yang, J. Ma, S. Ma, Y. Wang, and F. Li, "IMShell-Dec: Pay more attention to external links in powershell," in *Proc. IFIP Int. Conf. ICT Syst. Security Privacy Protection*, 2020, vol. 580, pp. 189–202.
11. C. Patsakis and C. Anargyros, "Analysing the Fall 2020 Emotet campaign," 2020, *arXiv:2011.06479*.
12. T. Seals, "Digitally signed bandook trojan reemerges in global spy campaign," ThreatPost, Nov. 30, 2020. https://threatpost.com/digitally-signed-bandook-trojan-spy-campaign/161676
13. T. Koziak, K. Wasielewska, and A. Janicki, "How to make an intrusion detection system aware of steganographic transmission," in *Proc. Eur. Interdisciplinary Cybersecurity Conf.*, Nov. 2021, p. 77–82.
14. "Steganography defensive initiative," McAfee. Accessed: Jun. 2022. https://www.mcafee.com/enterprise/it-it/downloads/free-tools/steganography.html
15. Extended Berkeley Packet Filter. [Online]. Available: https://ebpf.io (Accessed: Jun. 2022).
16. L. Caviglione, W. Mazurczyk, M. Repetto, A. Schaffhauser, and M. Zuppelli, "Kernel-level tracing for detecting stegomalware and covert channels in linux environments," *Comput. Netw.*, vol. 191, May 2021, Art. no. 108010, doi: 10.1016/j.comnet.2021.108010.
17. L. Caviglione, M. Gaggero, J. Lalande, W. Mazurczyk, and M. Urbański, "Seeing the unseen: Revealing mobile malware hidden communications via energy consumption and artificial intelligence," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 4, pp. 799–810, Apr. 2016, doi: 10.1109/TIFS.2015.2510825.

Luca Caviglione is a senior research scientist at the National Research Council of Italy, Institute of Applied Mathematics and Information Technologies, Genova, 16149, Italy. His research interests include security and networking. Caviglione received a Ph.D. in computer science from the University of Genova. Contact him at luca.caviglione@cnr.it.

Wojciech Mazurczyk is a university professor at the Institute of Computer Science, Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, 00—665, Poland, and at FernUniversität in Hagen, Hagen, 58084, Germany. His research interests include cybersecurity and communication networks. Mazurczyk received a D.Sc. (habilitation) in telecommunications from the Warsaw University of Technology. Contact him at wojciech.mazurczyk@pw.edu.pl.