

Wodax: Text entre les ombres

Joan Puigcerver Pérez

joapuipe@gmail.com

Text publicat sota els termes de la Llicència de Documentació Lliure de GNU.

16 de desembre de 2008

Índex

1	Introducció	2
1.1	<i>Esteganografia</i>	2
1.2	<i>Esteganografia</i> i criptografia	2
1.3	Conceptes fonamentals	3
1.3.1	Vocabulari	3
1.3.2	Imatges digitals	3
1.3.3	Caràcters digitals	4
2	Funcionament	5
2.1	Amagant caràcters	5
2.2	Extraient caràcters	6
2.3	Efectivitat i eficiència	7
2.4	Atacs	8
2.4.1	Força bruta	8
2.4.2	Comparança bit a bit	9
2.4.3	Reescriure la imatge	9
2.5	Millores de seguretat	9
2.5.1	Entropia	9
2.5.2	Xifratge	10
2.6	Restriccions	11
2.6.1	Grandària de la imatge amb relació a la del text	11
2.6.2	Longitud de la contrasenya	11
2.6.3	Imatges JPEG	11
3	Sobre el projecte en concret	11
3.1	Història	11
3.2	Instal·lació i ús	12
3.2.1	Instal·lació	12
3.2.2	Ús	12
3.3	Llibreries i codi font	13
3.4	Llicència	13
3.4.1	Del programa	13
3.4.2	De la documentació	13

1 Introducció

1.1 *Esteganografia*

El terme *esteganografia*¹ prové de les paraules gregues *steganos* (ocult) i *graphos* (escriptura) i és a l'antiga Grècia on apareix per primera vegada de la mà d'Heròdot i la seva *Història*[2]. Ací es pot llegir un fragment en el qual Histieu utilitza un mètode d'*esteganografia* primitiu per revoltar-se contra els perses.

Doncs com Histieu hagués volgut prevenir al seu parent que convenia rebel·lar-se, i no trobant mitjà segur per a posar-li l'avís puix que estaven els camins presos de part del rei, en tal dificultat havia rasurat a navalla el cap del criat que tenia de major satisfacció, haver-li marcat en ella amb els punts i lletres que li va semblar, va esperar després que li tornés a créixer el cabell, i crescut ja, haver-lo despatxat a Milet sense més encàrrec que dir-li de paraula que posat en Milet demanés de la seva part a Aristàgores que, tallant-li a navalla el pèl, li mirés el cap. Les notes gravades en ella significaven a Aristàgores, com vaig dir, que es revoltés contra el persa.

Però no fou fins el segle XVI quan Johannes Trithemius encunyà el terme *Steganographia* a la tècnica d'ocultar text i ho va fer en el seu llibre homònim[9].

No obstant això, al igual que succeí amb la criptografia (per exemple, la famosa màquina Enigma del Tercer Reich) i la majoria dels avanços tecnològics, l'*esteganografia* es potencià bàsicament com a arma de guerra. I per exemple, també en la II Guerra Mundial, s'insertien microfilms en els signes de puntuació dels missatges que contenien informació amagada. Així, una típica carta d'amor escrita per un soldat en el front, contenia en realitat la informació de moviments de l'exèrcit.

Ara, en plena època digital, l'*esteganografia* ja no usa eines tradicionals com la tinta invisible a partir de suc de limma sobre paper sinó que utilitza la informació digital, i té moltes més utilitats que les purament bèl·liques, com per exemple les marques d'aigua en documents digitals (fotografies digitals, vídeos, documents de text, etc) i de paper (alguns fabricants com HP i Xerox incorporen en alguns dels seus models aquestes opcions).

Fins i tot l'Agència Federal d'Investigació (FBI) o l'Agència Nacional de Seguretat (NSA) dels Estats Units, la tenen en compte per defensar-se de possibles atacs terroristes i han escrit diversos documents al respecte[3].

1.2 *Esteganografia* i criptografia

A sovint es confon l'*esteganografia* amb la criptografia, quan en realitat, tot i que els orígens de les dues tècniques són semblants i la finalitat també, els mètodes són ben diferents.

La criptografia² és la tècnica de transformar el contingut d'un missatge llegible en illegible (xifratge) de forma que únicament el receptor pugui ser capaç de fer el procés invers (desxiframent).

L'*esteganografia*³, però, és la tècnica d'amagar el contingut d'un missatge de forma que únicament el destinatari en sàpiga de la seva existència.

Així una tècnica criptogràfica amaga el contingut del missatge, però no el missatge en sí. Si un tercer detecta un missatge xifrat podrà no saber el contingut del missatge, però sens dubte sabrà de l'existència d'aquest. Amb una tècnica esteganogràfica, però, és el propi missatge qui s'amaga de forma que un tercer no en siga conscient de l'existència d'aquest. Aquesta diferència s'aprecia clarament en el problema dels

¹El mot "*esteganografia*" s'escriurà en cursiva durant tot el text per no tenir el significat que es recull en el diccionari del Institut d'Estudis Catalans, que es defineix com un sinònim de "criptografia". Les diferències s'explicaran en el punt 1.2 en la pàgina 2.

²Segons el diccionari del Institut d'Estudis Catalans: *f. [LC] [FL] [IN] Tècnica de conversió d'un text o d'unes dades en un missatge incompreensible per a la persona que no en posseeix la xifra o la clau.*

³Segons la Viquipèdia: *L'esteganografia és l'art i la ciència d'escriure missatges ocults de tal manera que només en coneix l'existència el destinatari previst.* Seria més correcte substituir "l'art i la ciència" per "la tècnica".

presoners descrit per G. J. Simmons[7].

Imaginem que hi ha dos presoners que volen planejar un pla per escapar-se de la presó, però tots els seus missatges passen primer per les mans d'un guardià. Com s'ho farien per comunicar-se sense que el guardià siga capaç d'adonar-se de les seves intencions? Evidentment, no poden simplement xifrar el missatge, ja que encara que el guardià no sàpiga el contingut d'aquest, això el farà sospitar. Així doncs, no és suficient amb una tècnica criptogràfica. Serà necessària una tècnica esteganogràfica per tal d'enganyar al guardià amb un fals contingut (contingut tapadora) i que no se n'adone del vertader missatge.

1.3 Conceptes fonamentals

Abans de començar a explicar el funcionament del programa, s'explicaran alguns conceptes bàsics sobre la representació d'imatges i caràcters digitals, i es presentarà un vocabulari elemental sobre *esteganografia*.

1.3.1 Vocabulari

Document tapadora Document en el que s'incrustarà el missatge que es pretèn ocultar.

Bit menys significatiu És la posició de bit en un nombre binari que té el valor més petit⁴.

Píxel El píxel és la unitat mínima de informació que forma una imatge digital⁵.

Model de color Un model de color permet representar els colors en forma numèrica, utilitzant tres o quatre valors, anomenats components cromàtics o canals. Existeixen diferents models de color com el RGB que utilitza els canals roig (**Red**), verd (**Green**) i blau (**Blue**), el CMYK que utilitza els canals cian (**Cian**), magenta (**Magenta**), groc (**Yellow**) i negre (**black**), o el tradicional RYB, utilitzat per la comunitat artística tot i no ser un model correcte, que utilitza el roig (**Red**), el groc (**Yellow**) i el blau (**Blue**)⁶.

Profunditat de color La profunditat de color fa referència a la quantitat de bits amb els que es representa un píxel. La profunditat de color és un indicatiu de la quantitat de colors diferents que es poden representar en un píxel. Les seves unitats de mesura són els bits per píxel (bpp) i les més típiques són 8 ($2^8 = 256$ colors), 16 ($2^{16} = 65536$ colors) o 24 ($2^{24} = 16777216$ colors)⁷.

Resolució d'imatge La resolució d'imatge indica la quantitat total de píxels que conté el mapa de bits d'una imatge. Es calcula a partir del nombre de píxels d'amplada (*width*) i d'altura (*height*) que té una imatge ($Resolució = Amplada \times Altura$). En les imatges capturades per càmeres fotogràfiques, la resolució sol ser de milions de píxels (Megapíxels)⁸.

1.3.2 Imatges digitals

Una imatge digital és la representació d'una imatge utilitzant bits. Existeixen dos tipus d'imatges (o gràfics) digitals, els gràfics basats en un mapa de bits (*raster graphics*) o els vectorials (*vectorial graphics*). La diferència rau en que en els primers, la imatge es representa com una matriu on cada posició és ocupada per un píxel utilitzant un model i una profunditat de color determinats i la segona està formada per objectes geomètrics (rectes, polígons, etc.) definits per atributs matemàtics de forma. A partir d'ací, sempre que es faça referència a una imatge serà en referència a una imatge digital basada en un mapa de bits, utilitzant el model de color RGB i amb una profunditat de color de 24 bits (que són les més habituals a dia d'avui).

⁴http://en.wikipedia.org/wiki/Least_significant_bit

⁵<http://en.wikipedia.org/wiki/Pixel>

⁶http://en.wikipedia.org/wiki/Color_model

⁷http://en.wikipedia.org/wiki/Color_depth

⁸http://en.wikipedia.org/wiki/Image_resolution

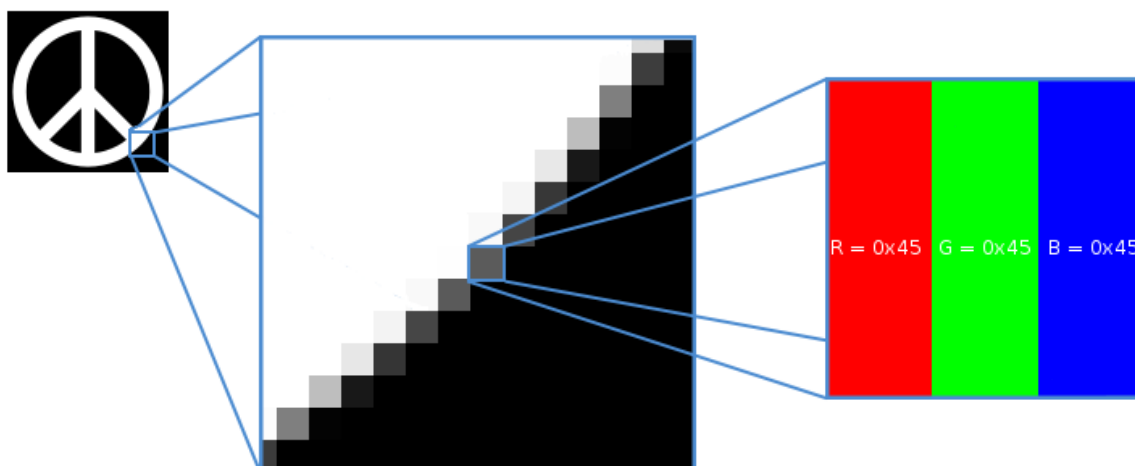


Figura 1: A l'esquerra la imatge completa, al centre els píxels d'una secció i a la dreta els valors dels canals per a un píxel.

Si es coneix la resolució de la imatge i la profunditat de color, es pot calcular el nombre de bits que ocupa la imatge, i per tant la quantitat d'informació contesa en aquesta. Per exemple, amb una imatge de 640×420 píxels de resolució amb una profunditat de color de 24 bits, la imatge té un total de $640 \times 420 \times 24 = 6451200 \text{ bits} = 6300 \text{ Kbits}$ (on $1 \text{ Kbits} = 2^{10} \text{ bits}$)⁹.

1.3.3 Caràcters digitals

En una computadora els caràcters es representen amb un valor numèric que depèn de la codificació emprada. Existeixen diverses codificacions, la més antiga és l'ASCII (*American Standard Code for Information Interchange*), però actualment se n'utilitzen moltes més com les ISO-8859-{1-16}, les pròpies de Microsoft Windows (Windows-125{0-8}), o les diferents variants de la Unicode.

La diferència fonamental entre cadascuna d'aquestes és el valor que utilitzen per representar un mateix caràcter i el nombre de bits que utilitzen per fer-ho. Tant la codificació Windows-1252¹⁰ com la ISO-8859-15¹¹ utilitzen 8 bits (1 byte) per representar un caràcter. Per tant hi han $2^8 = 256$ caràcters diferents que es poden representar utilitzant cadascuna d'aquestes codificacions. Però, per exemple, en la codificació Windows-1252, el caràcter "€" es codifica amb el valor decimal 128 i en la ISO-8859-15 es fa amb el 164.

D'altra banda la codificació ASCII¹² sols utilitza 7 bits per representar cada caràcter, de forma que hi han $2^7 = 128$ caràcters representables en la codificació (tot i que existeixen extensions de 8 bits per representar caràcters amb accents, per exemple, que són molt utilitzades).

Aquests 7 bits podrien semblar suficients per representar tot l'alfabet llatí i els dígitos aràbics, per exemple, però degut a la internacionalització dels programes informàtics, aquesta limitació pot suposar un problema, ja que hi han molts més alfabetos (japonès, rus, àrab, xinès, etc.), així com símbols matemàtics o lingüístics i hi hauria que utilitzar una codificació distinta en cada país. Aquest problema intenta resoldre'l l'estàndard Unicode¹³ i les seves codificacions UTF-8 i UTF-16 (de 8 i 16 bits de lon-

⁹Això no significa que una imatge PNG, per exemple, vaja a ocupar 6300 Kbits en memòria. La majoria de formats d'imatge tenen un espai reservat per a la capçalera on s'escriuen alguns atributs (com la resolució, la profunditat de color, el mode de color, etc.).

¹⁰<http://www.microsoft.com/globaldev/reference/sbcs/1252.msp>

¹¹<http://anubis.dkuug.dk/JTC1/SC2/WG3/docs/n404.pdf>

¹²<http://www.unicode.org/charts/PDF/U0000.pdf>

¹³<http://www.unicode.org>

gitud variable, respectivament).

No obstant això, en quasi tots els sistemes de codificació, els primers 128 valors coincideixen amb els de la codificació ASCII per motius de compatibilitat.

2 Funcionament

Una vegada entesos alguns conceptes fonamentals per entendre el procés que duu a terme Wodax, s'explicarà detalladament el funcionament d'aquest. Existeixen diverses estratègies per incrustar informació en documents tapadora, però Wodax utilitza la més comuna i intuïtiva: la inserció en el bit menys significatiu (*Least Significant Bit Insertion*). Es poden trobar diversos documents que expliquen el funcionament d'aquest mètode i d'altres com l'Emmascarament i filtratge i el basat en Transformacions ([5], [4]).

La idea és ben senzilla, s'inseriran els bits del missatge en els bits menys significatius de la imatge tapadora.

Per explicar el funcionament suposarem un missatge que utilitza la codificació ASCII que s'ocultarà en una imatge PNG (el programa treballa amb aquest format, en la secció 2.6.3 s'explicarà perquè no ho fa amb altres com el JPEG).

2.1 Amagant caràcters

S'explicarà primer un xicotet exemple de com amagar un caràcter en una xicoteta imatge i a partir d'aquest s'aprofundirà per explicar el funcionament sencer del programa. Imaginem que disposem d'una imatge, que utilitza el mode de color RGB i té una profunditat de color de 24 bits, composta pels píxels $P_0(R, G, B) = (45_{16}, 24_{16}, 52_{16})$ i $P_1(R, G, B) = (93_{16}, 0_{16}, 89_{16})$ i que volem ocultar el caràcter "A" en aquesta imatge.

En la secció 1.3.3 (en la pàgina 4) comentàvem que la codificació ASCII utilitzava 8 bits per representar cada caràcter. Així doncs, el caràcter "A" (valor 41_{16} en la taula ASCII) es pot veure de la següent manera bit a bit¹⁴.

C_3		C_2		C_1		C_0	
0	1	0	0	0	0	0	1

Taula 1: El caràcter "A" vist bit a bit.

Els píxels esmentats anteriorment es poden veure d'aquesta forma bit a bit.

P_0								
R	0	1	0	0	0	1	0	1
G	0	0	1	0	0	1	0	0
B	0	1	0	1	0	0	1	0

Taula 2: Pixel P_0 vist bit a bit.

P_1								
R	1	0	0	1	0	0	1	1
G	0	0	0	0	0	0	0	0
B	1	0	0	0	1	0	0	1

Taula 3: Pixel P_1 vist bit a bit.

¹⁴El bit més significatiu ocupa la posició a l'esquerre del byte.

El caràcter es separa en parelles de bits com s'ha indicat en la taula 1 i s'insereixen en els dos bits menys significatius de cada canal, començant pel roig. Quan tots els canals d'un píxel han sigut modificats, llavors es canvia de píxel i continua incrustant-se el caràcter, començant per la parella de bits del caràcter i el canal del píxel on s'havia quedat.

A continuació es mostra com quedarien els píxels de l'exemple anterior després d'amagar el caràcter "A". Els bits que han sigut modificats s'han marcat en **negreta**.

P_0								
R	0	1	0	0	0	1	0	1
G	0	0	1	0	0	1	0	0
B	0	1	0	1	0	0	0	0

Taula 4: Píxel P_0 vist bit a bit després d'amagar el caràcter "A".

P_1								
R	1	0	0	1	0	0	0	1
G	0	0	0	0	0	0	0	0
B	1	0	0	0	1	0	0	1

Taula 5: Píxel P_1 vist bit a bit després d'amagar el caràcter "A".

Si ara s'incrustés un nou caràcter, l'inserció d'aquest començaria pel canal verd (G) del píxel P_1 fins omplir aquest píxel i continuaria en el canal roig (R) d'un nou píxel P_2 que seria necessari.

De forma més genèrica, aquest és l'algorisme que s'utilitza per amagar un arxiu de text¹⁵.

Algorithm 1 Algorisme per amagar text en una imatge.

Require: $Image = (p_1, p_2, p_3, \dots, p_n) / \forall p_i \in Pixel$

Require: $Text = (c_1, c_2, c_3, \dots, c_n) / \forall c_j \in [0, 255]$

$i \leftarrow \alpha$

$j \leftarrow 0$

$C \leftarrow Red$

repeat

for $k = 0$ to 3 **do**

$p_i(C) \leftarrow \lfloor p_i(C) \div 2^2 \rfloor + (\lfloor c_j \div 2^{2k} \rfloor \bmod 2^2)$

if $C = Blue$ **then**

$i \leftarrow \beta$

end if

$C \leftarrow (C + 1) \bmod 3$

end for

$j \leftarrow j + 1$

until $c_j = EOF$

2.2 Extraient caràcters

Una vegada entès com s'incrusta un text en una imatge, es pot comprendre fàcilment com s'extreu aquest a partir de la imatge. Simplement s'ha de fer el procés invers. Recordem com havien quedat els píxels anteriors (les taules 4 i 5).

¹⁵El significat dels valors α i β s'explicarà en la secció 2.5.1, quan s'explicarà el concepte d'entropia per al programa. En aquest punt de l'explicació, podeu considerar $\alpha = 0$ i $\beta = i + 1$.

Ara s'extreuen els dos bits menys significatius de cada canal de cada píxel i s'agrupen formant grups de huit bits. A partir del canal roig (R) del píxel P_0 s'obté la parella de bits $C_0 = 01$, del canal verd (G) s'obtenen els bits $C_1 = 00$ i del canal blau (B) els bits $C_2 = 00$. Com que ja s'ha extret la parella de cada canal de P_0 , canviem de píxel i passem a P_1 per obtindre la parella de bits menys significativa del canal roig, i s'obté $C_3 = 01$.

C_3		C_2		C_1		C_0	
0	1	0	0	0	0	0	1

Taula 6: El caràcter "A" extret a partir de dos píxels.

El programa, doncs, recorre tots els píxels de la imatge extraient la parella menys significativa de bits de cada canal i agrupa aquestes parelles de quatre en quatre per formar caràcters de 8 bits. Veiem l'algorisme detalladament.

Algorithm 2 Algorisme per extreure un text amagat en una imatge.

Require: $Image = (p_1, p_2, p_3, \dots, p_n) / \forall p_i \in Pixel$

Ensure: $Text = (c_1, c_2, c_3, \dots, c_n) / \forall c_j \in [0, 255]$

$i \leftarrow \alpha$

$j \leftarrow 0$

$C \leftarrow Red$

repeat

$c_j = 0$

for $k = 0$ to 3 **do**

$c_j \leftarrow (p_i(C) \bmod 2^2) \times 2^{2k} + c_j$

if $C = Blue$ **then**

$i \leftarrow \beta$

end if

$C \leftarrow (C + 1) \bmod 3$

end for

$j \leftarrow j + 1$

until $c_j = EOF$

2.3 Efectivitat i eficiència

Ja s'ha explicat el funcionament del programa. La pregunta que s'ha de contestar ara és si aquest mètode és efectiu (si resol el nostre problema) i eficient (si ho fa amb un temps i amb un consum de memòria adequat).

El problema plantejat, recordem, era el d'aconseguir manipular una imatge per incrustar en l'interior un missatge de text i que una tercera persona no pogués adonar-se'n de l'existència d'aquest missatge.

Si es té qualsevol canal $C = XXXXX00_2 / X \in [0, 1]$, llavors, quan es modifiquen els seus dos bits menys significatius, els possibles valors són $C_0 = XXXXX00_2$ ($\Delta C = 0$), $C_1 = XXXXX01_2$ ($\Delta C = 1$), $C_2 = XXXXX10_2$ ($\Delta C = 2$) o $C_3 = XXXXX11_2$ ($\Delta C = 3$). Per tant, per a un canal, la variació màxima del seu valor és de 3 unitats. Tenint en compte que la diferència entre el valor màxim i mínim en un canal és 255 ($FF_{16} - 00_{16}$), la variació relativa del canal $\Delta C_R \leq 0.011764706$. I per tant, com que cada píxel està format per tres canals, la variació relativa de cada píxel serà $\Delta P_R \leq 0.035294118$.

A continuació, es presenta una demostració de l'efectivitat del programa amb l'exemple que s'ha utilitzat en l'apartat anterior. La diferència de color és inapreciable per a l'ull humà.

(0x45, 0x24, 0x52)	(0x93, 0x00, 0x89)
(0x45, 0x24, 0x50)	(0x91, 0x00, 0x89)

Figura 2: Demostració de l'efectivitat del programa amb l'exemple anterior. Dalt abans i baix després de modificar els píxels.

L'eficiència és més senzilla de discutir fins i tot. Veiem que tant en l'algorisme per amagar text com en el d'extreure'l, el bucle principal depèn linealment del nombre de caràcters del text. Per tant, el cost temporal d'ambdòs algorismes és $\Theta(C)$, on C és el nombre de caràcters del text.

Pel que fa al cost espacial, les úniques dades que necessiten ser carregades en memòria són la imatge on amagar o d'on obtenir el text i, per altra banda, el mateix text. Llavors, el cost espacial per als dos algorismes és $\Theta(C + P)$, on C és el nombre de caràcters del text i P el nombre de píxels de la imatge.

2.4 Atacs

L'*esteganografia* no és la tècnica definitiva per comunicar-se en secret amb una segona persona, té certes restriccions i debilitats¹⁶. Aquestes són sols algunes de les tècniques més senzilles que es poden utilitzar per invalidar aquesta tècnica, però és important saber que n'hi ha d'altres més sofisticades. Es poden consultar alguns documents per la Internet com el de Niels Provos i Peter Honeyman[6],

2.4.1 Força bruta

Un atac per força bruta és l'atac més senzill que es pot aplicar a qualsevol tècnica esteganogràfica o criptogràfica, i en certa manera no hi ha cap mesura que pugui inhibir-lo. Però, a la pràctica aquest atac, si està ben dissenyat el sistema, resulta *computacionalment* impossible de dur a terme.

Si l'atacant coneix el píxel on comença a amagar-se el text, podria fàcilment fer-se amb el text contingut en aquest repetint el procés descrit en la secció 2.2, ja que l'algorisme és conegut per aquest¹⁷.

Si no sap la posició inicial, però els píxels es modifiquen amb un determinat increment de posició a partir de la inicial, llavors, provant amb diferents posicions inicials i diferents increments de posició podria també fer-se amb un missatge amagat. El temps que tarda en fer-ho depèn la grandària de la imatge, tindria que provar per a cada possible posició inicial, tots els possibles increments de posició, que anirien des de 1 fins a la mateixa grandària de la imatge. Això suposa un cost $O(P^2)$.

Si la imatge és suficientment gran, aquesta tasca es faria a la pràctica impossible. En la secció 2.5.1 s'explicarà una mesura per millorar la defensa contra aquest atac.

¹⁶La tècnica per explotar aquestes debilitats s'anomena *esteganoanàlisi*, al igual que el criptoanàlisi a la criptografia.

¹⁷La seguretat d'un determinat algorisme criptogràfic o *esteganogràfic* mai pot estar basada en la ignorància de l'atacant sobre aquest. Aquest enunciat es coneix com el Principi de Kerckhoff.

2.4.2 Comparança bit a bit

Si l'atacant compta amb la imatge original on s'ha amagat el text, obtenir-lo és una tasca quasi trivial. Podem comparar bit a bit les dues imatges (l'original i la sospitosa d'haver estat modificada) i conèixer quines són les parelles de bits que difereixen i agrupar-les per tal d'obtenir el missatge complet.

Podrien haver parelles de bits que al modificar-se quedaren igual que en la imatge original (en l'exemple que es plantejava en la secció 2.1 aquesta situació es donava al amagar el parell de bits C_1 i C_2), però ara un atac per força bruta seria possible de dur a terme, ja que s'haurien eliminat moltes possibilitats.

2.4.3 Reescriure la imatge

Recordem el problema dels descrit per Simmons en el que els dos presoners havien d'evitar que el guàrdia se n'adonés de que estaven passant-se missatges. Em discutit una forma que tenen aquests presoners de comunicar-se secretament bastant eficaç, però que passa si el guàrdia sospita d'ells? Si el guàrdia omplira de *soroll* els missatges que s'intercanvien els dos presoners, el contingut d'aquests arribaria alterat de l'un a l'altre i sense sentit. Aquest és el principi d'aquest atac.

Si es sap que una determinada imatge té quelcom ocult (i es pot saber emprant una funció *hash*), únicament omplint amb bits aleatoris totes les parelles de bits menys significatius dels canals de cada píxel, s'assegura de que el contingut secret de la imatge queda destruït. L'atacant no coneixerà el contingut del missatge, però el destinatari tampoc.

De fet, aquest mètode fou proposat per un estudiant de la Universitat de Northeastern per tal d'implementar-lo en els servidors de correu de forma que, quan s'enviés algun determinat tipus d'arxiu (imatges, música, vídeos, etc.), s'apliqués el mètode, destruint així la possible informació amagada[1].

2.5 Millores de seguretat

Tal i com s'havia dit en l'anterior apartat, hi ha certes mesures que es poden aplicar per fer més segur l'algorisme (i que de fet s'han aplicat en el programa).

2.5.1 Entropia

Aquesta és una mesura que s'implementà per intentar fer ineficaços els atacs per força bruta i per comparació de bits. El concepte del programa prové del concepte físic d'entropia, la mesura del desordre d'un determinat sistema físic. En el nostre cas, podríem entendre la imatge com un sistema on l'entropia es refereix al desordre dels píxels modificats en la imatge. Així, si els píxels es modifiquen de forma contigua o seguint una determinada seqüència lògica, la imatge té un grau d'entropia baix, però si per contra els píxels es modifiquen aleatòriament, podem dir que la imatge té un grau d'entropia elevat.

Evidentment, no podem modificar els píxels completament de forma aleatòria ja que es necessita determinar quins són els píxels que s'han modificat per tal de poder extreure la informació posteriorment. Per això s'utilitza un generador de nombres pseudo-aleatoris.

Una generador d'aquest tipus genera una seqüència de nombres a partir d'una determinada "llavor". Els nombres generats seran diferents per a cada llavor, però sempre seran els mateixos per a una mateixa llavor donada.

En el Wodax, la llavor s'obté a l'aplicar-li una funció *hash* de 32 bits a la contrasenya que l'usuari indica (aquesta contrasenya també servirà en un futur per xifrar el missatge abans d'amagar-lo en la imatge). Tot i que amb una funció *hash* de 32 bits hi ha nombroses col·lisions¹⁸, això no té "massa"¹⁹ importància ja que serveix com una barrera que pot dificultar (però no impossibilitar) els atacs.

¹⁸Una "col·lisió" *hash* ocorre quan per a dos claus diferents, la funció *hash* torna el mateix valor.

¹⁹En realitat té moltíssima importància ja que amb una altra contrasenya es podria desxifrar el missatge!

El procediment que es segueix per determinar la posició inicial a partir d'on començaran a ocultar-se els caràcters del text, és el següent.

Algorithm 3 Selecciona el primer píxel a modificar

```
 $X \leftarrow \text{Random}() \bmod \text{Width}$   
 $Y \leftarrow \text{Random}() \bmod \text{Height}$ 
```

On *Width* i *Height* són l'amplada i l'altura en píxels de la imatge i *Random()* es una funció generadora de nombres pseudo-aleatoris.

Quan s'ha de seleccionar un nou píxel perquè ja s'han modificat tots els canals de l'actual, es segueix el següent procediment.

Algorithm 4 Selecciona el nou píxel a modificar

```
 $X' \leftarrow (X + (\text{Random}() \bmod \text{Entropy}) + 1) \bmod \text{Width}$   
if  $X' \leq X$  then  
     $Y \leftarrow (Y + 1) \bmod \text{Height}$   
end if  
 $X \leftarrow X'$ 
```

Els valors (X, Y) fan referència a la posició del píxel que es modifica. Ara s'entén el significat dels valors α i β que apareixien en les seccions 2.1 i 2.2. El valor de α ve determinat pel resultat del primer dels dos algorismes anteriors i el de β per el segon.

Amb aquests dos procediments s'aconsegueix introduir aquest grau de desordre que es pretenia per dificultar un possible atac per força bruta. El valor de l'entropia s'obté a partir de la longitud de la contrasenya, així que com més llarga siga aquesta major grau de desordre s'aconsegueix. Opcionalment, també es pot forçar al programa que utilitze un valor determinat²⁰.

2.5.2 Xifratge

En primer lloc dir que el Wodax no té integrada una funcionalitat que permeti xifrar el text al mateix temps que s'oculta en una imatge. El motiu és que l'autor no coneix suficientment bé cap mètode de xifratge realment segur per implementar-lo (dos possibles candidats són el Tiger i el Whirlpool). Wodax és un programa amb una finalitat púrament didàctica, així que poc aprendria el seu autor si simplement utilitzés una llibreria d'una tercera persona per incorporar aquesta funcionalitat. Fet aquest aclariment, s'ha decidit incloure aquest concepte dintre de l'apartat "Millores de seguretat" ja que ha de ser una característica fonamental i en la que ja s'està treballant ara mateixa.

La diferència fonamental entre la criptografia i l'*esteganografia*, com ja s'ha comentat abans, és que amb la primera es pretén amagar el contingut d'un missatge, però no el fet de que s'està transmetent aquest. L'objectiu de la segona és precisament el contrari, amagar l'existència del propi missatge. Però imaginem que el guàrdia dels dos presoners que descrivia Simmons se n'adona de què aquests estan tramant alguna cosa i vol saber què. Si per qualsevol motiu, el guàrdia descobreix el missatge amagat, els dos presoners estan perduts.

El que deuen fer els dos presoners, és acordar un mètode de xifratge i d'aquesta forma, si el guàrdia descobreix el contingut amagat en el missatge tapadora (en el cas del Wodax, el missatge contingut en la imatge), abans de poder llegir-lo haurà de desxifrar-lo (i en cas de que s'elegisca un bon mètode de xifratge, és poc probable que això passe). Aquesta és la idea, en lloc d'amagar en la imatge el text original, amagar el text xifrat.

²⁰Com es pot extreure a partir del seu ús en els algorismes, el valor de l'entropia haurà de ser major estrictament que 1.

2.6 Restriccions

A continuació s'expliquen algunes restriccions que té el programa o algunes característiques que es deurién tenir en compte a l'hora d'amagar missatges.

2.6.1 Grandària de la imatge amb relació a la del text

Com que la posició on començar a amagar el text s'elegeix aleatòriament, aquesta podria ser la posició d'un dels últims píxels de la imatge. Si això succeeix, el programa continua escrivint en la imatge per el principi, de forma que la imatge forma una espècie de cercle on no hi ha ni principi ni final.

Per tant s'ha d'elegir amb compte la grandària de la imatge per assegurar-se de que tot el text que es pretén amagar té cabuda en la imatge. De la forma en que Wodax amaga un caràcter en la imatge, és necessiten $\frac{4}{3}$ píxels per amagar 1 caràcter. Llavors, si s'han d'amagar c caràcters, el nombre de píxels p de la imatge hauria de ser en principi $p \geq \lceil c \times \frac{4}{3} \rceil$.

Això seria així si els píxels es modificaren de forma contigua, però amb el concepte d'entropia que s'ha explicat anteriorment, això canvia, ja que en la imatge quedaran píxels que no s'aprofitaran i per això haurà de ser més gran.

El valor de l'increment de posició amb una entropia ϵ està en l'interval $[1, \epsilon]$. Així, el nombre de píxels p necessaris per amagar un text de c caràcters i utilitzant un valor per a l'entropia ϵ , seran $\lceil c \times \frac{4}{3} \rceil \leq p \leq \lceil (\frac{1}{3} + \epsilon) \times c \rceil$.

2.6.2 Longitud de la contrasenya

La restricció es aquest aspecte és que en el procés de compilació del programa es defineix una longitud màxima per a la contrasenya (per comoditat). Si compileu vosaltres mateixa el programa, podeu definir la longitud que trobeu oportuna, si descarregueu un binari ja compilat des de la pàgina web, la longitud maxima predeterminada es 500 (en aquest cas, la grandària de la contrasenya pot ser qualsevol valor $1 \leq l \leq 500$).

En la secció 3.2 s'explicarà com definir la longitud de la contrasenya en el procés de compilació.

2.6.3 Imatges JPEG

Podria sorgir el dubte de per què s'ha utilitzat el format PNG per treballar amb les imatges i no un altres molt més estès com podria ser el JPEG. Hi ha dos motius pels quals no s'ha utilitzat el JPEG²¹, un tècnic i l'altre ètic.

El primer motiu és que el JPEG és un format de compressió d'imatges amb pèrdua de qualitat i els mètodes per amagar text en una imatge JPEG i després extreure'l correctament són més complicats que substituir els bits menys significatius en la imatge. Però hi ha tècniques d'*esteganografia* per poder utilitzar aquest tipus d'imatge [10].

El segon motiu és que hi ha certa incertesa²² relacionada amb el format JPEG i patents a les que estan subjectes alguns dels algorismes que utilitza. Per això s'ha utilitzat el PNG, un format molt estès en la Web també.

3 Sobre el projecte en concret

3.1 Història

La idea del projecte sortí després de llegir el llibre *Cryptonomicon* de Neal Stephenson[8]. El llibre havia fet interessar-me per el tema de la criptografia i l'*esteganografia* i llegint sobre el segon em sorgí la idea

²¹<http://www.w3.org/Graphics/JPEG/itu-t81.pdf>

²²http://en.wikipedia.org/wiki/JPEG#Potential_patent_issues

de fer un programa *estegano*-criptogràfic per tal de posar en pràctica algunes de les idees sobre les quals havia llegit. Ací està el resultat (part d'ell, perquè això encara no ha acabat).

El nom “Wodax” prové de “Shadow”. Quan tenia llesta la primera versió del programa estava buscant un bon nom, original i senzill i que a la vegada estigués relacionat amb el món de la criptografia i l'*esteganografia*. De sobte, em vingué al cap la frase “S'ocultava entre les ombres...” (no em pregunteu perquè... Segurament, dec tenir un trauma infantil). “Shadow” era un bon nom per al meu programa, ja un pot amagar coses fàcilment en la ombra, igual que el programa en les imatges. Però el nom no era massa original.

Llavors, vaig canviar les lletres “Sh” per la “X” perquè en català la “X” al principi d'una paraula té la mateixa pronunciació en alguns casos que la “Sh” en anglès. Finalment, li vaig pegar la volta al nom i... “*tukun' tish!*”: Ja tenia nom per al programa.

Una altra curiositat sobre el programa és l'ordre que segueix el nombre de la versió. El nombre de la versió vindrà donat per un nombre real de la forma $x.y$ (x és la part entera i y la fraccionària), on $x_i = y_{i-1}$ i $y_i = \frac{\varphi^i - (-\varphi)^{-i}}{\sqrt{5}}$, partint de $x_0 = 0$ i $y_0 = 1$. El que això significa és que en la versió $i0$ la part entera es desplaça a la fraccionària i s'introdueix el terme i -ésim de la Successió de Fibonacci a la part fraccionària. Les versions seran les següents: **0.1**, **1.1**, 1.2, 2.3, 3.5, 5.8, 8.13, etc (marcades en negreta les que ja han estat publicades).

Aquesta forma de numerar les versions és una picada d'ull a la numeració que segueix el projecte \TeX , que a partir de la seva tercera versió inclou una xifra decimal més de manera que el número de la versió tendeix a π (quan Donald Knuth, el creador, morí es fixarà el número de la versió a π)²³.

3.2 Instal·lació i ús

3.2.1 Instal·lació

Per compilar el programa hi ha prou amb executar l'ordre `make` en la carpeta on hages descarregat el programa.

Pots modificar les opcions de compilació editant l'arxiu `makefile` que acompanya als arxius de programa. A continuació es mostra una taula que descriu la funció de cada opció.

Opció	Funció	Valor per defecte
CXX	Compilador	g++
FLAGS	Opcions del compilador	-Wall -pedantic -O3
INSTALL_PATH	Ruta de la instal·lació	/usr/bin
MAX_PASS	Longitud màxima de la contrasenya	500

Taula 7: Opcions de compilació, les seves funcions i els seus valors per defecte.

Una vegada compilat pots executar l'ordre `make clean` per netejar el directori i eliminar els arxius temporals.

Per instal·lar el programa en la carpeta indicada en la variable `INSTALL_PATH` executa l'ordre `make install` i per eliminar-lo `make uninstall`.

3.2.2 Ús

Per amagar text executa `wodax --hide (-h) image.png [opcions]` i per extreure el text ocult en una imatge `wodax --seek (-s) image.png [opcions]`. Les opcions disponibles venen resumides en aquesta taula.

²³http://en.wikipedia.org/wiki/Software_versioning#TeX

Opció	Funció [Mode = Hide o Seek]	Valor per defecte
--output	Nom de la imatge amb el text ocult [H]	Imatge d'entrada
--file	Fitxer per amagar en o extreure d'una imatge [H,S].	Eixida/Sortida estàndard
--entropy	Valor de l'entropia [H,S].	Longitud de la contrasenya.

Taula 8: Opcions d'execució del programa.

3.3 Llibreries i codi font

Les llibreries extra que s'han utilitzat per construir el programa, a banda de les llibreries estàndard de C i C++, són la `libpng`²⁴ i un embolcall en C++ per a aquesta, anomenat `png++`²⁵. La primera té una llicència Zlib i la segona una modificació de la llicència BSD, ambdues compatibles amb la llicència GPLv3 sota la que es publica Wodax.

L'última versió del codi font la podeu trobar en la font de programari del projecte²⁶ o en la meua pàgina web²⁷.

3.4 Llicència

3.4.1 Del programa

Wodax és programari lliure: podeu redistribuir-lo i/o modificar-lo sota els termes de la Llicència Pública General de GNU publicada per la Free Software Foundation, ja sigui la versió 3 de la Llicència, o (a la seva elecció) qualsevol versió posterior.

Wodax es distribueix amb l'esperança que sigui útil, però SENSE CAP GARANTIA; ni tan sols la garantia implícita MERCANTIL o d'APTITUD PER A UN OBJECTIU PARTICULAR. Consulteu els detalls de la Llicència Pública General de GNU per a més informació.

Haurieu de rebre una còpia de la Llicència Pública General de GNU junt amb aquest programa. En cas contrari, consulteu <http://www.gnu.org/licenses/>.

3.4.2 De la documentació

Copyright (c) 2008 Joan Puigcerver Pérez. Es concedeix permís per copiar, distribuir i/o modificar aquest document sota els termes de la Llicència de Documentació Lliure de GNU, Versió 1.3 o qualsevol altra versió posterior publicada per la Free Software Foundation; sense Seccions Invariants ni Textos de Coberta Davantera ni Textos de Coberta Del Darrere.

Aquest document deuria anar acompanyat d'una còpia de la llicència. En cas contrari consulteu <http://www.gnu.org/licenses/>.

Referències

- [1] BERTOLINO, K. Spy vs. spy. *IEEE Spectrum* (2008). <http://spectrum.ieee.org/aug08/6593>.
- [2] HERÒDOT. *Els nou llibres de la Història*, vol. V. 440aC.
- [3] KESSLER, G. C. An overview of steganography for the computer forensics examiner. Tech. rep., Computer and Digital Forensics Program, Champlain College, 2004. http://www.fbi.gov/hq/lab/fsc/backissu/july2004/research/2004_03_research01.htm.

²⁴<http://www.libpng.org/>

²⁵<http://www.nongnu.org/pngpp/>

²⁶<http://svn.assembla.com/svn/wodax/>

²⁷<http://www.jpucgserver.net>

- [4] KRENN, J. Steganography and steganalysis, 2004. <http://www.krenn.nl/univ/cry/steg/article.pdf>.
- [5] NEIL F. JOHNSON, S. J. Steganography: Seeing the unseen. *IEEE Computer Journal* (1998). <http://www.jjtc.com/pub/r2026.pdf>.
- [6] NIELS PROVOS, P. H. Detecting steganographic content on the internet. Tech. rep., Center for Information Technology Integration, University of Michigan, 2001. <http://www.citi.umich.edu/techreports/reports/citi-tr-01-11.pdf>.
- [7] SIMMONS, G. J. The prisoner's problem and the subliminal channel. Tech. rep., Sandia National Laboratories, Albuquerque, 1983. <http://dsns.csie.nctu.edu.tw/research/crypto/HTML/PDF/C83/51.PDF>.
- [8] STEPHENSON, N. *Cryptonomicon*. Avon, 1999. <http://en.wikipedia.org/wiki/Cryptonomicon>.
- [9] TRITHEMIUS, J. *Steganographie: Ars per occultam Scripturam animi sui voluntatem absentibus aperiendi certu*. 1500.
- [10] YEUAN-KUEN LEE, L.-H. C. Secure error-free steganography for jpeg images. Tech. rep., Department of Computer and Information Science, National Chiao Tung University, 2003. <http://www.csie.mcu.edu.tw/~yklee/Publications/SEFSJ.pdf>.