Tools steganografici

a cura di:
Pasquale Mieo
Silvestro Giardiello

Etimologia della parola

La parola steganografia deriva dal greco:

stèganos: nascosto

gràfein: scrivere

Differenze con crittografia

- Crittografia:

  Ha lo scopo di nascondere il contenuto di un messaggio
- Steganografia:

  Ha lo scopo di nascondere l'esistenza del messaggio

Uso della steganografia

- In molte situazioni l'uso della sola crittografia non è sufficiente
- Esempio: soldato che scambia messaggi cifrati col governo di un paese ostile

Problema dei prigionieri

Simmons (1983)

- Alice e Bob sono in prigione sorvegliati dal guardiano Willie
- Alice e Bob vogliono escogitare un piano per fuggire
- Se Willie scopre che Alice e Bob scambiano messaggi cifrati li mette in isolamento ed il loro piano fallisce
- Problema: trovare un metodo per nascondere il testo cifrato in un testo apparentemente innocuo

Steganografia classica

- Erodoto
- Griglie di Cardano
- Cifre nulle
- Inchiostri invisibili
- Micropunti fotografici

Schema di base

La steganografia presuppone l'esistenza di due messaggi:

- Messaggio segreto
- Messaggio contenitore

Modelli steganografici

Steganografia iniettiva

Modelli steganografici

Steganografia generativa

Altra classificazione

- Steganografia sostitutiva
- Steganografia selettiva
- Steganografia costruttiva

Steganografia sostitutiva

Rumore

- Si produce ogni volta che c'è un processo di conversione da analogico a digitale.
- Deriva dall'imperfezione del dispositivo di acquisizione (scanner, scheda sonora, scheda di acquisizione video)
- Influenza i bit meno significativi delle codifiche digitali

Rumore

- Sostituendo i bit meno significativi (influenzati dal rumore) con i bit del messaggio segreto, il file sarà modificato in modo impercettibile ai sensi umani.
- Gli stessi bit inseriti potranno essere confusi con rumore.
- Il più delle volte il nemico non possiede il file originale per effettuare un confronto

File bitmap (.BMP)

Immagine = Matrice di MxN pixel

Codifica in formato RGB

Caso 24 bit: 3 byte in sequenza per ogni pixel

File bitmap (.BMP)

Osservazioni:

- Ogni livello di colore primario può avere valori tra 0 e 255
- In totale ci sono 2^{24} =16777216 colori possibili per ogni pixel
- Un'immagine 640x480 occupa 640x480x3=921600 byte

Applicazione steganografia

Sia un certo pixel codificato come segue:

11100001 00000100 00010111

Possiamo inserire 3 bit del messaggio segreto. Se volessimo inserire 110 il nostro pixel sarà trasformato così:

11100001 00000101 00010110

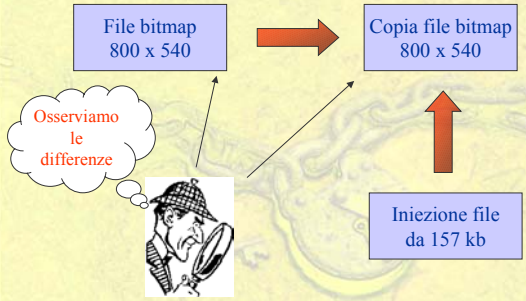
Applicazione steganografia

Le operazioni su ogni singolo byte possono essere 3:

1) Lo si lascia invariato
2) Gli si aggiunge 1
3) Gli si sottrae 1

Ad occhio nudo le variazioni sono impercettibili!

Esperimento



Esperimento

35	35	2E	35	38	33	35	37	32
32	36	33	30	33	2E	2F	31	2C
34	38	30	3F	46	3D	4C	54	46
42	4D	3E	38	3F	35	3C	3B	31
3F	3C	35	3D	3F	36	3A	3B	3C
3A	3B	3A	3C	43	3A	3F	47	3D
40	44	3E	40	43	3F	39	3C	3B
3C	3E	40	44	51	55	49	56	57
47	4E	50	4D	52	52	50	52	4B
4B	4D	44	47	49	44	41	46	45
43	4B	47	48	4C	45	43	45	3E

File originale

35	35	2E	35	38	32	35	36	32
33	37	33	31	32	2F	2E	31	2D
34	39	31	3E	46	3C	4D	55	46
42	4C	3F	39	3E	35	3D	3B	30
3F	3C	35	3C	3E	36	3A	3A	3C
3A	3A	3B	3C	43	3A	3F	47	3D
41	44	3F	41	42	3E	39	3D	3B
3D	3E	41	45	51	54	49	57	56
47	4F	51	4D	52	53	51	53	4A
4B	4C	45	46	49	44	40	47	44
42	4A	47	49	4D	45	43	44	3F

Copia con iniezione del file segreto

Dimensione messaggio

Per ogni pixel possiamo nascondere 3 bit, quindi:

Dimensione messaggio segreto (in byte) = (M x N x 3) / 8

In un file 640x480 possiamo nascondere (640 x 480 x 3) / 8 =115200 byte

Dimensione messaggio

Possibilità di aumentare lo spazio disponibile per il messaggio segreto:

Utilizzare non uno, ma i 2, 3 o 4 bit meno significativi

Peggioramento qualità, rumore più evidente

File wav

Formato 44100 Hz, 16 bit, stereo

Nella fase di digitalizzazione è generata una stringa di bit ogni 1/44100 di secondo

La lunghezza della stringa generata è di 16 bit

Sono generate due stringhe di bit alla volta, una per il canale destro ed una per il sinistro

File wav

Stesso concetto dei file bitmap: sostituzione bit meno significativi.

File wav

Esempio: file wav 44100 Hz, 16 bit, stereo di un minuto.

Dimensione file =16 bit x 44100 Hz x 60 sec x 2 = 84762000 bit = 10366 Kb

Spazio per il file nascosto (usando i due bit meno significativi) = 84762000 bit / 16 bit x 2 = 10595250 bit = 1293 Kb

Problemi

- I file bmp e wav sono abbastanza ingombranti
- Non sono usati molto su Internet
- Il solo fatto di scambiare file bmp o wav potrebbe destare sospetti
- Soluzione: utilizzare formati più diffusi

Formato jpg

La struttura dei file jpg impedisce di utilizzare una semplice tecnica sostitutiva

Soluzione?

~~Iniettare delle informazioni in un file bmp e poi convertirlo in jpg~~

La compressione JPG ha la tendenza a preservare le caratteristiche visive dell'immagine piuttosto che la sequenza di pixel originaria

Formato jpg

La struttura dei file jpg impedisce di utilizzare una semplice tecnica sostitutiva

Soluzione?

Iniettare le informazioni nei coefficienti di Fourier ottenuti dalla prima fase di compressione

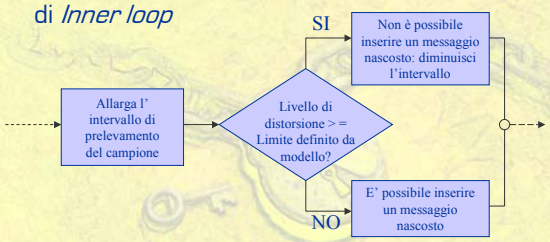
Formato mp3

Anche in questo caso non è possibile iniettare il messaggio segreto nel file wav e poi comprimere

Soluzione?

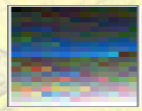
Formato mp3

Si inserisce il messaggio segreto nella fase di *Inner loop*



Formato gif

- Formato grafico molto utilizzato per i siti web perché poco ingombrante
- Si basa su una palette di 256 colori
- Un file gif è una sequenza di puntatori alla palette (uno per ogni pixel)



Formato gif

Come iniettare un file segreto in un file gif?

1) Acquisire immagine 
2) Decrementare il numero di colori ad un numero inferiore a 256 con opportuno algoritmo che limita la perdita di qualità
3) Si converte in gif riempiendo la palette con colori molto simili a quelli rimasti

Formato gif

Dopo un'operazione di questo tipo ogni pixel potrà essere rappresentato alternativamente con il colore originale o con il relativo colore aggiunto simile.

In presenza di alternative possiamo nascondere un'informazione!

Esempio: 2 alternative => si può nascondere un bit (se il bit è 0 scegliamo la prima, se è 1 la seconda)

Problema

- E' molto semplice scrivere un programma che analizzi la palette ed individui sottoinsiemi di colori simili e quindi la probabile presenza di un messaggio nascosto.

Formato gif: altra soluzione

Osservazioni:

- Un'immagine gif può essere rappresentata in 256! modi diversi perché ci sono 256! permutazioni per la stessa palette.
- Siamo in presenza di 256! alternative => possiamo codificare log 256! = 1683 bit indipendentemente dalle dimensioni dell'immagine.

Altri formati

Particolari software hanno esteso i concetti base della steganografia anche ad altri formati come PDF, HTML (wbStego) o MID, AVI, MPEG (Datamark Technologies)

Steganografia e sicurezza

Regole a cui attenersi:

- Mai usare file pubblici o facilmente accessibili come file contenitore
- Mai usare più volte lo stesso file contenitore (conviene generarne ogni volta uno nuovo con un dispositivo di acquisizione) 
- Distruggere il file contenitore originale dopo l'iniezione 

Steganografia e sicurezza

Principale difetto della steganografia sostitutiva:

- Le sostituzioni possono alterare le caratteristiche statistiche del rumore nel media utilizzato.
- Se il nemico possiede un modello del rumore può testare se i file sono conformi al modello: se non lo sono probabilmente c'è un messaggio nascosto.

Steganografia e sicurezza

E' difficile costruire un modello del rumore, ma in alcuni casi un attacco basato su un modello funziona.

Soluzione?

Steganografia selettiva e steganografia costruttiva

Steganografia selettiva

- Ha valore puramente teorico, non è utilizzata in pratica
- Idea: procedere per tentativi fino a quando non si verifica una certa condizione

Esempio steg. selettiva

- Sia *f* una funzione del file contenitore che dà 1 se il numero di bit 1 è dispari e dà 0 se il numero di bit 1 è pari
- Vogliamo codificare 0
- Acquisiamo un'immagine 
- Se ha un numero pari di bit 1, OK, altrimenti acquisiamo un'altra immagine fino a soddisfare questa richiesta

Steganografia selettiva

Pregi:

- Il file contenitore contiene il messaggio segreto senza essere stato modificato!

Difetti:

- Richiede troppo tempo per essere applicata nella pratica
- Permette di nascondere una quantità modesta di informazione

Steganografia costruttiva

- Opera come la steganografia sostitutiva, ma nella sostituzione tiene conto di un modello del rumore
- Il falso rumore introdotto deve attenersi il più possibile al modello

Problemi

- Non è facile costruire un modello del rumore
- E' sempre possibile che venga costruito un modello più accurato
- Se il modello cade nelle mani del nemico, egli potrebbe analizzarlo per cercarne i punti deboli: si regalerebbe al nemico uno strumento di attacco molto efficace!

Sistema steganografico ideale

Principio di Kerchoff:

La sicurezza del sistema deve basarsi sull'ipotesi che il nemico abbia piena conoscenza dei dettagli di progetto e implementazione del sistema stesso; la sola informazione di cui il nemico non può disporre è una sequenza (corta) di numeri casuali - la chiave segreta – senza la quale, osservando una canale di comunicazione, non deve avere neanche la più piccola possibilità di verificare che è in corso una comunicazione nascosta.

Sistema steganografico ideale

- La semplice steganografia sostitutiva non aderisce al principio di Kerchoff

Soluzione?

Fase di preelaborazione del messaggio segreto

Sistema steganografico ideale

Soluzione ovvia: cifrare il messaggio segreto prima di iniettarlo.

Problemi:

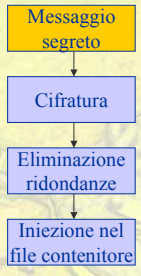
- Il file cifrato può comunque essere estratto da chiunque
- Un crittoanalista esperto può facilmente riconoscere un file prodotto da un programma di crittografia convenzionale

Sistema steganografico ideale

Soluzione: usare programmi capaci di eliminare tutte le informazioni diverse dal blocco di dati cifrati (ridondanze) e ricostruirle successivamente.

Una volta estratto il messaggio segreto è difficile distinguere testo cifrato da rumore

Sistema steganografico ideale



Software utilizzato

Software	Formati supportati	Algoritmi di cifratura
Steganos 3 Security Suite	WAV, BMP	AES
S-tools 4	WAV, BMP e GIF	IDEA, DES, 3DES, MDC
Jsteg shell 2.0	JPG	RC4-40
Mp3Stego	MP3	?
Gif-it-up 1.0	GIF	?

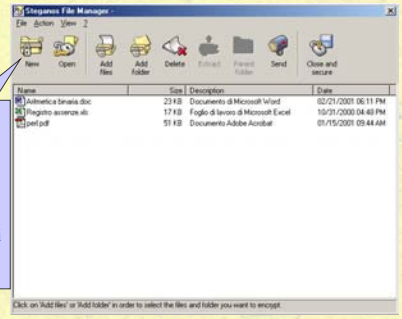
Steganos 3 Security Suite

Programmato da:	DEMCOM - Francoforte (D) CenturionSoft – Washington (USA)
Versione:	3
Data:	Primi mesi 2001
Piattaforma/Sistema operativo:	PC IBM compatibili/Windows 95 e superiori
Licenza:	Shareware (30 giorni di uso gratuito)

Steganos 3 Security Suite

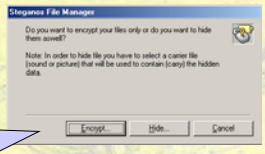
Steganos File Manager

Cliccando su New abbiamo la possibilità di compilare l'elenco dei file da nascondere



Steganos File Manager

Possiamo scegliere di cifrare solo i file o di nasconderli anche in un file contenitore



Steganos File Manager

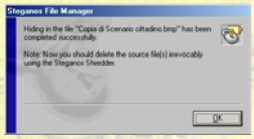
Selezione del file contenitore

Steganos File Manager

Inserimento password

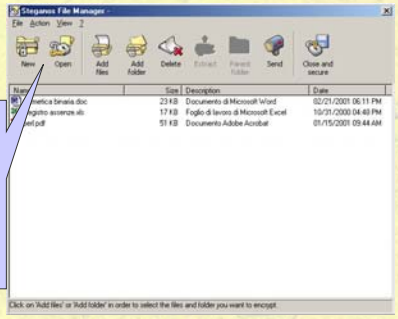
Steganos File Manager

Inserimento completato!



Steganos File Manager

Per estrarre basta cliccare su Open, selezionare il file ed inserire la password



S-tools

Programmato da:	Andy Brown
Versione:	4
Data:	1996
Piattaforma/Sistema operativo:	PC IBM compatibili/Windows 95 e superiori
Licenza:	Freeware

S-tools

Interfaccia basata su drag'n'drop

S-tools (iniezione)

1) Si trascina il file contenitore da una finestra di un file manager nella finestra principale di S-Tools.
2) Si trascina il file segreto sul file contenitore
3) Si sceglie la passphrase e l'algoritmo di cifratura; si può scegliere tra IDEA, DES, Triple DES o MDC, tutti utilizzati in modalità CFB.
4) A questo punto viene generato il nuovo file contenente il messaggio segreto ed è possibile salvarlo.

S-tools (estrazione)

1) Si trascina il file contenitore da una finestra di un file manager nella finestra principale di S-Tools.
2) Si seleziona *Reveal* cliccando col tasto destro del mouse sul file contenitore trascinato.
3) Si sceglie la passphrase e l'algoritmo di cifratura.
4) Se la passphrase è giusta compare un elenco dei file nascosti ed è possibile salvarli.

Jsteg Shell

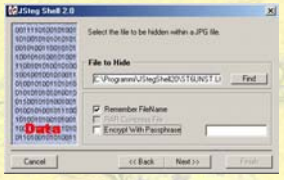
Programmato da:	Derek Upham e John Korejwa
Versione:	2.0
Data:	?
Piattaforma/Sistema operativo:	PC IBM compatibili/Windows 95 e superiori (escluso Windows 2000)
Licenza:	Freeware

Jsteg Shell (iniezione)

Schermata principale: scelta tra *Hide* e *Extract*

Jsteg Shell (iniezione)

Selezione file da nascondere e passphrase



Jsteg Shell (iniezione)

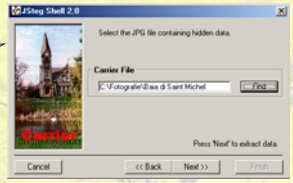
Selezione file contenitore ed opzioni di compressione

Jsteg Shell (iniezione)

Salvataggio file di output

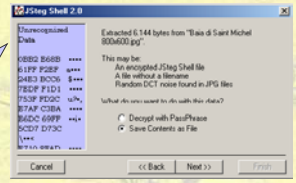
Jsteg Shell (estrazione)

Selezione file contenitore



Jsteg Shell (estrazione)

Salvataggio file con eventuale inserimento passphrase



Mp3Stego

Programmato da:	Fabien Petitcolas (Computer Laboratory, Cambridge (UK))
Versione:	1.1.15
Data:	Agosto 1998
Piattaforma/Sistema operativo:	PC IBM compatibili / Dos, Windows e Linux
Licenza:	Freeware

Mp3Stego

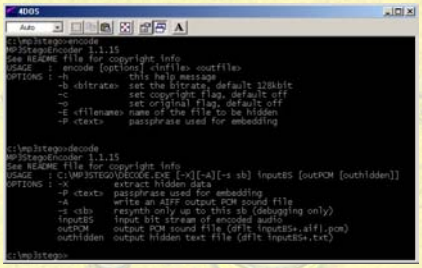
- Interfaccia a linea di comando
- Encode:

```
encode -E data.txt sound.wav
sound.mp3
```

- Decode:

```
decode -X sound.mp3
```


Mp3Stego

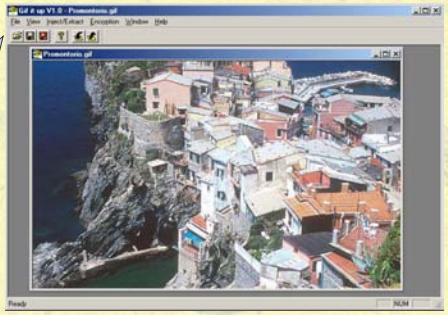


Gif-it-up

Programmato da:	NelsonSoft (Università del Galles)
Versione:	1.0
Data:	1998
Piattaforma/Sistema operativo:	PC IBM compatibili / Windows 95 e superiori (escluso Windows 2000)
Licenza:	Freeware

Gif-it-up (iniezione)

Per iniettare un file basta aprire il file contenitore, scegliere se estendere i colori a tutta la palette, selezionare il comando *Inject* ed inserire la passphrase



Gif-it-up (estrazione)

L'operazione di estrazione *(Extract)*, dopo aver aperto il file contenitore semplicemente richiede la password e ci fa salvare il file nascosto estratto.