La Steganografia

La stenografia, nell'ambito dell'informatica, si riferisce principalmente a una disciplina diversa dalla stenografia tradizionale (che si occupa della scrittura veloce). Nel contesto informatico, la stenografia è una tecnica utilizzata per nascondere informazioni all'interno di altri dati, spesso in modo da risultare invisibile o difficile da rilevare. Questo tipo di stenografia è noto come steganografia.

Steganografia Informatica

La steganografia informatica è l'arte e la scienza di nascondere messaggi o dati all'interno di file digitali come immagini, audio, video o altri tipi di media. L'obiettivo è mantenere segreta l'esistenza del messaggio nascosto, piuttosto che criptare il contenuto del messaggio stesso (che è invece l'obiettivo della crittografia).

Come Funziona

La steganografia si basa sulla modifica di dati digitali in modo che i cambiamenti siano impercettibili all'occhio o all'orecchio umano. Ad esempio, una tecnica comune è la steganografia delle immagini. In questo caso, piccoli bit di informazioni vengono nascosti nei bit meno significativi di ogni pixel dell'immagine. Anche se questi bit vengono modificati, la differenza risultante nell'immagine è così piccola da essere invisibile.

Un esempio:

Un'immagine JPEG potrebbe essere usata per nascondere un messaggio di testo modificando leggermente i valori dei pixel. Anche se un computer può rilevare questi cambiamenti, una persona che guarda l'immagine non noterà alcuna differenza.

Lo stesso concetto può essere applicato a file audio, video o di testo. La steganografia non si limita a file multimediali, ma può essere implementata in vari tipi di dati digitali.

Applicazioni della Steganografia nell'Informatica

Sicurezza e Comunicazioni Segrete:

Spionaggio e intelligence: La steganografia è utilizzata per trasmettere messaggi segreti senza attirare l'attenzione. Per esempio, agenti di intelligence possono nascondere informazioni sensibili in immagini o file audio e inviarli senza destare sospetti.

Comunicazioni sicure: Alcuni individui utilizzano la steganografia per evitare la sorveglianza o censura, nascondendo informazioni critiche all'interno di file innocui.

Protezione dei Diritti d'Autore (Watermarking):

Una delle applicazioni legittime della steganografia è il watermarking digitale, utilizzato per proteggere i diritti d'autore. In questo caso, informazioni come il nome del proprietario del copyright o i dettagli della licenza possono essere nascosti all'interno di un file multimediale. Questi marchi d'acqua digitali possono essere utilizzati per identificare violazioni di copyright, anche se l'opera viene distribuita senza autorizzazione.

Autenticazione dei File:

La steganografia può essere utilizzata per garantire che un file digitale non sia stato alterato. Ad esempio, un messaggio nascosto può essere incorporato in un documento o in un'immagine, e in seguito verificato per assicurarsi che non ci siano state modifiche non autorizzate.

Resistenza alla Censura:

In contesti in cui la censura è una preoccupazione, la steganografia può essere utilizzata per diffondere informazioni o per aggirare i blocchi informatici. Gli attivisti potrebbero nascondere messaggi all'interno di immagini o video per evitare che vengano intercettati da regimi oppressivi.

Sicurezza nelle Reti di Computer:

La steganografia può anche essere utilizzata in ambito di sicurezza delle reti per nascondere comunicazioni all'interno del traffico di rete. Ad esempio, un attacco informatico sofisticato potrebbe includere comandi nascosti nel traffico di rete che appaiono innocui a un osservatore casuale.

Sfide e Problemi della Steganografia Informatica

La steganografia, pur essendo una potente tecnica, presenta alcune sfide:

Rilevabilità: Anche se l'obiettivo è nascondere i dati, gli analisti esperti possono utilizzare strumenti specializzati per rilevare la presenza di dati nascosti. Tecniche di steganalysis vengono sviluppate per scoprire tali dati.

Capacità limitata: La quantità di dati che può essere nascosta dipende dal supporto utilizzato. Ad esempio, non è possibile nascondere un lungo documento in una piccola immagine senza degradare la qualità visibile dell'immagine.

Uso improprio: La steganografia può essere sfruttata da malintenzionati per attività illegali, come la distribuzione di contenuti illeciti o la trasmissione di dati relativi a malware.

Differenze tra Steganografia e Crittografia

Mentre la crittografia nasconde il contenuto del messaggio trasformandolo in una forma illeggibile a chi non possiede la chiave di decodifica, la steganografia nasconde l'esistenza stessa del messaggio. Spesso, queste due tecniche vengono combinate per aumentare il livello di sicurezza.

Conclusione

La steganografia nell'informatica è una potente tecnica per nascondere informazioni, con applicazioni che spaziano dalla protezione dei diritti d'autore alla comunicazione sicura. Tuttavia, poiché può essere usata per scopi sia legittimi che illeciti, la steganografia rimane una tecnologia ambigua e spesso monitorata da esperti di sicurezza informatica.

Esempio di codice:

```
from PIL import Image


def text_to_binary(message):
    # Converte il messaggio di testo in binario
    return ''.join([format(ord(char), '08b') for char in message])


def binary_to_text(binary):
    # Converte il messaggio binario in testo
    binary_chunks = [binary[i:i+8] for i in range(0, len(binary), 8)]
    return ''.join([chr(int(chunk, 2)) for chunk in binary_chunks])


def hide_message(image_path, message, output_image_path):
    # Apri l'immagine
    img = Image.open(image_path)
    binary_message = text_to_binary(message) + '1111111111111110'  # Codice speciale di terminazione


    data = iter(img.getdata())


    new_data = []
    for pixel in data:
        new_pixel = list(pixel)
        for i in range(3):  # Modifica solo i canali RGB, ignorando l'eventuale canale alfa
            if len(binary_message) > 0:
                new_pixel[i] = new_pixel[i] & ~1 | int(binary_message[0])
                binary_message = binary_message[1:]
```

```
            new_data.append(tuple(new_pixel))

        img.putdata(new_data)
        img.save(output_image_path)
        print("Messaggio nascosto con successo nell'immagine!")

def reveal_message(image_path):
    img = Image.open(image_path)
    binary_message = ''

    for pixel in img.getdata():
        for i in range(3):  # Leggi solo i canali RGB
            binary_message += str(pixel[i] & 1)
            # Codice di terminazione del messaggio: 16 '1' seguiti da uno '0'
            if binary_message.endswith('1111111111111110'):
                return binary_to_text(binary_message[:-16])

    return "Nessun messaggio nascosto trovato."

# Esempio di utilizzo
image_path = 'immagine_originale.png'
output_image_path = 'immagine_con_messaggio.png'
message = "Questo è un messaggio segreto."

# Nasconde il messaggio nell'immagine
hide_message(image_path, message, output_image_path)

# Rivela il messaggio nascosto dall'immagine
print("Messaggio nascosto:", reveal_message(output_image_path))
```

Spiegazione:

text_to_binary(message): Converte ogni carattere del messaggio di testo in una rappresentazione binaria di 8 bit.

binary_to_text(binary): Converte una stringa binaria in un testo leggibile.

hide_message(image_path, message, output_image_path):

Apre l'immagine specificata.

Converte il messaggio di testo in binario e aggiunge un codice di terminazione speciale (1111111111111110).

Itera attraverso i pixel dell'immagine e modifica i bit meno significativi (LSB) di ogni canale RGB con i bit del messaggio binario.

Salva l'immagine modificata con il messaggio nascosto.

reveal_message(image_path):

Apre l'immagine e legge i bit meno significativi dei pixel.

Ricostruisce il messaggio in binario e lo converte in testo fino a trovare il codice di terminazione (1111111111111110).

Note:

Il codice di terminazione 1111111111111110 serve per indicare la fine del messaggio nascosto.

È importante notare che questo metodo può essere utilizzato solo con immagini non compresse o debolmente compresse, come PNG. Non funzionerà bene con formati compressi con perdita, come JPEG, perché la compressione può alterare i dati nascosti.

Considerazioni di Sicurezza

Questo codice è un esempio basilare di steganografia. In un'applicazione reale, potrebbe essere necessario utilizzare tecniche più avanzate per garantire che il messaggio sia più difficile da rilevare e che non possa essere facilmente estratto senza autorizzazione.