Secret Sharing with Multi-cover Steganographic Audio Files

**洪胤勛 吳坤熹**

 **國立暨南國際大學 資訊工程學系**

**{s108321019,Solomon}@ncnu.edu.tw**

摘要

祕密共享（secret sharing）[1, 2]為一種分享資料的技術，它會把資料拆成若干份sharing，並且把sharing分給不同的人。若要回復原本的資料，則需要湊齊大於或等於一定門檻(threshold)數量的sharing，否則不能還原出原本的資料。本論文將探討如何安全的把這些sharing分送給不同的人，並導入隱寫術，將sharing藏在聲音中，讓這些sharing在不易被察覺的情況下發送給不同的人。

1. 前言

隨著資安的議題日益重要，許多加解密的原理與工具更顯得它們的重要性。有些機密性的資料，不能夠直接赤裸裸的流通在網路上，因此有些人選擇給資料做了加密，防止資訊外洩。但在某些情況下，僅僅只是做了加密還是不夠的。比如在一個被監控的網路環境中，或是網路會流經不受信任的地點，當你的封包攜帶了加密的資料，管理者人員通常能很快發現此封包異常，因而此封包可能就此被丟棄又或是被選出來破解。為了避免封包被遺棄或是被別人嘗試破解，有人會選擇把機密的資料藏在一些不容易發現的地方，像是圖片或是聲音，甚至是封包的表頭，此方法又稱隱寫術（steganograph）。

Secret sharing 在資安的領域也相當重要，它與加解密不同，secret sharing為一種分享資料的技術，在這種架構下，它解決了祕密資料掌握在單一個人手上的問題。比如需要多人授權的密碼，像是金庫的密碼，老闆可能會有緊急狀況，無法親自到金庫，因此需要部下幫他開起，但如果把密碼只交給一個人，那將非常危險，因此理想狀況是需要多人才能開啟；又或是階層式授權的密碼，給經理和主管有開啟的權力，但也不能讓單一個經理或是單一個主管直接擁有密碼，而是讓一個經理和一個主管，或是三個主管決定要打開金庫時，才能得出密碼。這些問題都可以靠secret sharing來解決。

Secret sharing同時也解決了機密資料掌握在單一個人手上的兩個問題。第一為資料的穩健性（robustness），在單一個資料的架構下，當系統被駭客入侵，資料被竊取走，那就表示機密一定被駭客知曉；但在secret sharing的架構下，即使駭客竊取走單一一份資料，他也無法得知祕密到底為何，他必須去找到一定數量的資料才有辦法還原出原始的機密資料。第二為資料的可靠性（reliability），在單一資料的架構下，資料若不小心遺失或是被駭客破壞掉了，那就再也無法取得祕密；但在secret sharing的架構下，即使遺失一份或損壞一份資料仍然可以靠其他的資料得以復原。

Secret sharing的做法在1979年相繼被Adi Shamir[1]和George Blakley[2]發表。而他們的解法又常分別被稱為Shamir’s Secret Sharing與Blakley’s Secret Sharing， Shamir’s Secret Sharing使用多項式來達成目的，而Blakley’s Secret Sharing使用有限域的幾何空間。雖然兩者都有threshold的場景，但Blakley’s Secret Sharing較複雜且沒效率[1]，因此本論文將著重在Shamir’s Secret Sharing。

本論文除了使用Shamir’s Secret Sharing外，為了讓資料不輕易被察覺，還使用隱寫術，把產生的sharing藏在音檔裡面，並且派送至不同目的地。

1. 研究動機

Secret sharing與一般的單一祕密架構不同，它會拆分成很多不同的sharing，因為要收集到多份的sharing才能解開祕密，這使得它要被破解更困難；因為要破壞掉多份的sharing，才能使祕密真正被破壞掉，這讓外人要被破壞它更艱辛。以上兩個原因，讓secret sharing比起單一祕密的架構更加強韌，更可以信任。

但也因為祕密會被拆分成多個sharing，這讓傳遞和保存sharing時，需要花費額外的資源來儲存和傳送這些sharing。如果sharing赤裸裸地在封包裏頭，那麼可能在這些sharing到達目的地前，就會偵測並刪除了，這並不是我們想要的結果。因此，本論文研究如何讓sharing能夠在安全、不易被察覺的情況下抵達目的地。

1. 相關研究
2. Securing matrix counting-based secret-sharing involving crypto steganography[3]

這篇的作者使用了matrix-based secret sharing[4]，一種由counting-based secret sharing[5]改進而來的secret sharing方法。它把由祕密產生的sharing，使用三原色（RGB）的圖片，並使用兩種不同的隱寫術。第一種使用LSB（Least Significant Bit），第二種使用了DWT（Discrete Wavelet Transform）。

1. Secret sharing with multi-cover adaptive steganography[6]

這篇的作者提出了multi-cover，跟傳統只藏在單一張圖片不同，作者把 sharing隱藏到不同的圖片裡。藏的地方越多，單一張圖片儲存的資訊越少，因此也更不容易被發現。

1. Cyber warfare: steganography vs. steganalysis[7]

在此篇論文裡，作者針對圖片LSB的隱寫術做了隱寫分析。他發現，通常在一張圖片上，一個點的LSB會與其鄰近點的LSB相似。因此只要抽取出整張圖片的LSB，行成LSB-plane，並針對此做比較，就可以很輕易發現這張圖片裡面是否有藏訊息，如果有藏訊息，LSB-plane將會有某些部分是不連續的。

1. 研究目標

本論文將使用secret sharing來提升祕密的穩健性和可靠性。除了secret sharing外，同時使用隱寫術來隱藏sharing，讓sharing不容易被發現。

在Cyber warfare[7]這篇文章裡，作者提到使用LSB把資訊藏在圖片裡面是件危險的事，因為此方法很容易被破解。為此，本論文將使用聲音取代圖片來進行隱寫術，聲音除了沒有LSB連續的問題外，聲音也不像圖片一樣，能夠靜態地仔細觀察整張圖片；聲音只能在連續播放的過程中聆聽。

除了聲音之外，本論文還設計了一個server與多個clients的架構。server把已經把藏在音檔的sharing送到多個不同client，如圖1.所示，此方式讓想竊取機密的駭客，必需要能夠聽取不同連線，才能有破解的一絲希望，因此更難去執行。



圖1. Server與client

1. 技術背景
2. Shamir’s Secret Sharing

Shamir’s Secret Sharing的演算法共有兩個步驟。第一步驟是將原本的機密資料D拆成*n*份sharing，並決定需要幾份以上(*k*)的sharing才能還原，故*k* ≤ *n*，並以*k*的值來決定一個多項式的最高冪為*k*–1，而零次項的值則為*D*。

第二步則從這*n*份sharing中湊得*k*份sharing，並將它還原為原本的祕密，若湊得的份數少於*k*份，則無法還原。拆成*n*份sharing與需要*k*份sharing才能組回去原本祕密的情境，又稱為(*k*,*n*) threshold。

以下將以threshold為3，sharing數 *n*=5，祕密*D*=34為例。在Shamir’s Secret Sharing的第一步驟中，將產生一個多項式，由於*k*為3，因此需要產生的多項式的最高次方為2，而零次項的值為34，其它次項的值則可隨機產生，以公式(1)為例，

$$F\left(x\right)=4x^{2}-21x+34(1)$$

由於n為5，因此需要在這平面上隨機找5個點，F(2)=8、F(3)=7、F(4)=14、F(5)=29、F(6)=52。

而在Shamir’s Secret Sharing第二步驟中，使用了拉格朗日差值法（Lagrange Interpolation）。如果收集到了3個sharing，F(3)=7、F(5)=29、F(6)=52，並代入拉格朗日差值法(2)，即可得到以下式子，簡化後即可得出(1)，得出*D*=34。

$$\begin{array}{c}\begin{array}{c}F\left(x\right)=7\frac{\left(x-5\right)\left(x-6\right)}{\left(3-5\right)\left(3-6\right)}+29\frac{\left(x-3\right)\left(x-6\right)}{\left(5-3\right)\left(5-6\right)}\\+52\frac{\left(x-3\right)\left(x-5\right)}{\left(6-3\right)\left(6-5\right)}(2)\#\end{array}\end{array}$$

1. 聲音取樣

在音檔中，以下幾個是控制聲音格式的重要參數，channel 、sample、frame 、sample rate、sample format。

1. channel代表著總共有幾個聲道，例如若是要輸出在電話的話筒，那麼channel數就設定為1，如果要輸出在有左右聲道的耳機，那麼channel數就設定為2。
2. sample為儲存聲音的一個最基本單位，其大小由sample format控制，用來表示聲音當下的狀態。
3. frame為所有聲道，在某個時間點的sample所成集合。如果是單聲道，那在某個時間點的一段frame裡就只有一個sample。如果是雙聲道，那在某個時間點的一段frame裡就只會有兩個sample。
4. sample rate為聲音的採樣頻率，單位為赫茲。sample rate越高，代表聲音的品質越好，但也代表資料量越大。電話的頻率為八千赫茲（Hertz，簡寫為Hz），而CD（Compact Disc）的音質大都是44100赫茲。
5. sample format為呈現每個sample的數值，最常見的有8-bit、16-bit和32-bit，bit數越多，能呈現聲音的範圍越大，聲音越準確。
6. Least Significant Bit

Least Significant Bit簡稱LSB。指的是在二進位的數字中，最小的位數。由於最小位數的更動，代表著更動後的值只會相差-1、0或1，不容易被發現，因此經常被拿來當作隱寫術隱藏資料的位置。

1. Peak signal-to-noise ratio[8]

Peak signal-to-noise ratio簡稱PSNR，用來測量訊號的雜訊比，單位為分貝。其定義如(3)

$$PSNR=10log\_{10}\left(\frac{MAX^{2}}{MSE}\right)(3)$$

以計算聲音的PSNR來說，MAX值為sample的最大值。而MSE（Mean Square Error）代表原始聲音與加入雜訊後的聲音之間的均方差，其定義如(4)

$$MSE=\sum\_{i=1}^{n}\left(C\_{i} - S\_{i}\right)^{2}(4)$$

其中*C*為原始聲音，*S*代表加入雜訊後的聲音。

1. 實驗架構
2. 決定參數

在此實驗架構中，將會有server和client兩個角色的架構。首先server需要決定好祕密*D*和(*k*,*n*)，並由ssss[9]產生*n*份sharing後，等待client的連線。

1. server錄音

當有client連上server時，server則可以決定是否要傳送資料。當server決定要開始傳送資料時，server則會開始錄音。在本實驗中，將會以sample rate為8000、channel為1，並用sample format為unsinged int8來錄製5~10秒鐘的聲音，用來模擬一則語音訊息。

1. 隱寫術

錄完音後，會把*n*份sharing的其中一份藏進錄音檔裡。使用的方式是用是用LSB藏在sample裡，本實驗使用的format為unsinged int8，所以每個sample就為一個byte，也就是一個byte可以藏入一個bit。如圖2.所示，sharing為**1-0bf53ca7dd**，長度為12 bytes。在隱藏資料前，我們需要先用一個byte來表示所藏資料的大小，表示共有12個字元。再把12用一個byte來表示，為00001100。之後把00001100這8個bits藏進前8個sample中。接著把sharing用ASCII來表示，也就是一個英文字元用8個bit表示，所以一個字元會分別藏在8個sample裡，以上述的範例來說，則需要96個sample來藏。長度連同sharing，全部共會用到8+96=104個sample。



圖2. 隱藏範例

1. 包進WAV檔

把已經藏好的sample，包成WAV檔，並使用TCP把此WAV檔送到client。之後，server又回到第二步，直到所有sharing被領取完畢。

1. client解析封包

client收到WAV檔後，會先把所有的sample從WAV檔取出來，並從前八個sample取得sharing的長度。取得長度之後，就能知道總共要再取幾個sample，並得出sharing。

1. 取回祕密

在*n*個client中，他們手上都會有不同的sharing。只要聚集*k*份以上的sharing，就能透過ssss[9]得出祕密D。

1. 實驗結果

在實驗的步驟二裡，我們模擬了語音訊息傳送的情境，使用format為unsinged int8，channel為1，並由ssss[9]以(*k*,*n*)為(3,5)、產生bytes為12的sharing來藏入104個sample裡。以下server將分別測量錄音5~10秒，每次固定一個秒數，發送5個sharing，並使用(3)與(4)比較它們之間的MSE和PSNR，因為使用的format為unsinged int8，所以MAX的值為255。結果如表I、圖3.和圖4.所示

表I

|  |  |  |  |
| --- | --- | --- | --- |
| 秒數 | Sample數 | MSE | PSNR |
| 5 | 40000 | 0.0012450 | 77.197667 |
| 6 | 48000 | 0.0011041 | 77.704725 |
| 7 | 56000 | 0.0009178 | 78.512879 |
| 8 | 64000 | 0.0008000 | 79.113535 |
| 9 | 72000 | 0.0007333 | 79.482942 |
| 10 | 80000 | 0.0006575 | 79.963869 |



圖3. MSE比較



圖4. PSNR比較

1. 結論

本實驗用來隱藏sharing的音檔皆是由server接收到client的連線後才開始錄音的，每次傳出去的音檔都是隨機的。因此，當有多個sharing要發送到不同地方時，每次發送的音檔都是不一樣，可以以此來降低攻擊者的疑慮。

發送音檔的目的地皆不相同，因此要在多個接收端去攔截信號，且能夠偷到threshold，*k*數量的音檔，又為攻擊者加上一層阻礙。

1. 未來展望

在聲音傳送的部分，用TCP來傳送仍不是最常見的形式，通常即時性的語音、視訊不會使用TCP來傳送，而是使用UDP或RTP來傳送。

但UDP和RTP並不像TCP一樣，能夠確保封包完整的到達。為了確保重要的sharing能夠傳達到目的地，因此將考慮加入錯誤更正碼在sharing中，讓server與client即使在網路環境不好的情況下，封包傳送錯誤時，也能夠自動修正，從而還原出正確的sharing，而不是整段重傳，吸引攻擊者的注意。

參考文獻

[1] A. Shamir, "How to share a secret," *Communications of the ACM,* vol. 22, no. 11, pp. 612-613, 1979.

[2] G. R. Blakley, "Safeguarding cryptographic keys," in *Managing Requirements Knowledge, International Workshop on*, 1979: IEEE Computer Society, pp. 313-313.

[3] F. Al-Shaarani and A. Gutub, "Securing matrix counting-based secret-sharing involving crypto steganography," *Journal of King Saud University-Computer and Information Sciences,* 2021.

[4] S. Porwal and S. Mittal, "A threshold secret sharing technique based on matrix manipulation," in *AIP Conference Proceedings*, 2020, vol. 2214, no. 1: AIP Publishing LLC, p. 020020.

[5] A. Gutub, N. Al-Juaid, and E. Khan, "Counting-based secret sharing technique for multimedia applications," *Multimedia Tools and Applications,* vol. 78, no. 5, pp. 5591-5619, 2019.

[6] H.-D. Yuan, "Secret sharing with multi-cover adaptive steganography," *Information Sciences,* vol. 254, pp. 197-212, 2014.

[7] H. Wang and S. Wang, "Cyber warfare: steganography vs. steganalysis," *Communications of the ACM,* vol. 47, no. 10, pp. 76-82, 2004.

[8] M. Tayel, A. Gamal, and H. Shawky, "A proposed implementation method of an audio steganography technique," in *2016 18th international conference on advanced communication technology (ICACT)*, 2016: IEEE, pp. 180-184.

[9] B. Poettering. "ssss." <https://linux.die.net/man/1/ssss> (accessed.