Szteganográfia

Házi feladat

Készítette: Bertók Zsófia (A6MHQV)

1. Mi is az a szteganográfia?

A szteganográfia görög eredetű szó, amelynek jelentése „rejtett üzenetet írni”. A szteganográfia egy olyan tudományág, amely bizonyos üzenetek elrejtésével foglalkozik. A kriptográfia lényege, hogy a továbbított üzenet létét nem titkoljuk, viszont az üzenet elolvasására mégsem képes akárki, mivel az üzenetet titkosított formában továbbítjuk, amelyet csak a megfelelő titkos kulccsal rendelkező fél tud visszafejteni. Szteganográfia esetén a kriptográfiával ellentétes módon a továbbítandó üzenetnek nem a tartalmát rejtjük el, hanem magát a létét titkoljuk. Az angol „security by obscurity” kifejezés azt jelenti, hogy a biztonságot valaminek a titkosságával garantáljuk, amely a szteganográfiára is teljességgel illik, ahol is az információközlés ténye marad rejtett. A szteganográfiával titkosított üzenetek megfejtésével foglalkozó tudományágat szteganalízisnek nevezzük.

1.1 A szteganográfia története

A szteganográfia egy nagy múltra visszatekintő tudományág. A kifejezést ugyan csak 1499-ben használta először Johannes Trithemius Steganographia című könyvében, azonban a módszert már időszámításunk előtt is ismerték. A legelső írásos emlékek Hérodotosz feljegyzései (Kr. e. 440), amelyben több olyan történetet is elmesél, melyek során bizonyos üzenetek elrejtése központi szerepet játszott. Az egyik ilyen történet szerint Demeratusz úgy figyelmeztette a spártaiakat Xerxész (perzsa király) szándékairól, hogy egy viasztáblára a viaszréteg alá rejtette el az üzenetet, amelynek köszönhetően az üzenet nem volt észrevehető külső szemlélő számára, így sikeresen eljuthatott Spártába. Egy másik történet arról szól, hogy Hisztiaiosz úgy üzente meg Arisztagorasznak, hogy szervezzen felkelést a perzsák ellen, hogy leborotváltatta egy rabszolgája haját, majd a csupasz fejbőrére tetováltatta az elrejteni kívánt üzenetet. Miután a szolga haja kinőtt, útnak indította Arisztagoraszhoz, aki a rabszolga haját leborotválva értesült Hisztiaiosz rejtett üzenetéről.

A történelem során még számos esetben előfordult a szteganográfia valamilyen formában, például kedvelt módszer volt üzenetek elrejtésére a láthatatlan tinta használata, amelyet a II. világháború során is használtak. A rejtett tintával írt szöveg előcsalogatásához általában valamilyen speciális folyadékra vagy esetleg felmelegítésre volt szükség. A mai értelemben vett szteganográfia kialakulása a kriptográfiához hasonló módon a digitális korszak kezdetére tehető.

1.2 Alapfogalmak

Napjainkban a szteganográfia alatt digitális üzenetek elrejtését értjük, mégpedig digitális tartalmakba történő rejtését. A szteganográfiai eljárások megértéséhez az alábbi kifejezések fogalmát célszerű tisztázni:

- *Hordozó médium (cover media)*: a rejtéshez használt médium, azaz az a fájl, amelybe a titkolni kívánt információt rejteni szeretnénk
- *Sztego médium (stego media)*: a rejtett adatot tartalmazó médium, azaz az a fájl, amelyben már megtalálható az elrejtett információ
- *Sztego kulcs (stego key)*: az adatrejtés során használt kulcs

Az utóbb említett sztego kulcsra akkor van szükség, ha az adatelrejtési mechanizmus során valamilyen titkos kulcsot is használunk, azaz a szteganográfiai algoritmusunk paraméterezhető egy kulccsal.

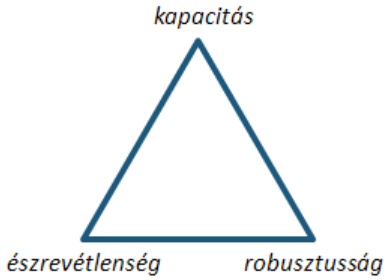
1. ábra: Az szteganografikus rejtés folyamata

2. A szteganográfiai algoritmusok jellemző paraméterei

Az egyes szteganográfiai algoritmusokat különböző mértékekkel szoktuk jellemezni, melyek a következőek:

- *kapacitás*: a hordozó médium mekkora része használható fel adatrejtés során
- *észrevétlenség*: mennyire észrevehető az elrejtett információ léte
- *robusztusság*: a sztego médiumon végrehajtott transzformációknak mennyire áll ellen az elrejtett információ
- *komplexitás*: a szteganográfiai algoritmus komplexitása (idő- és tárkomplexitás)

A kapacitás, észrevétlenség és robusztusság feltételek valamilyen szinten egymásnak ellentmondó feltételek. A köztük lévő kapcsolatot szokás az ún. kényszerháromszöggel ábrázolni.



2. ábra: A kapacitás, robusztusság és észrevétlenség tulajdonságok viszonya egymáshoz (kényszerháromszög)

2.1 Az algoritmusok csoportosítása

Az algoritmusokat egyrészt bekategorizálhatjuk aszerint, hogy a fentebb említett tulajdonságok közül melyiket milyen mértékben elégíti ki, emellett azonban egy másik csoportosítási módszert jelenthet az, hogy az adatrejtés feloldása hogyan történik. Ez alapján három kategóriát különböztethetünk meg, amelyek a következők:

- *Blind*: az adatrejtés feloldásához szükség van az eredeti (hordozó) médiumra is
- *Non-blind:* a feloldáshoz csak a sztego médium kell, nincs szükség az eredeti médiumra
- *Semi-blind:* az eredeti médium ugyan nem kell az adatrejtés feloldásához, de szükség van valamilyen további információra is az sztego médiumon kívül

Az egyes algoritmusok csoportosítását elvégezhetjük az alapján is, hogy milyen típusú médiumba, fájlba rejtjük az elrejtendő információt. Adatot elrejthetünk szöveges anyagokban, képekben, hangfájlokban, videófájlokban, de akár futtatható állományokban is.

3. Adatrejtés szöveges anyagokban

A szöveges anyagokba történő adatrejtés valamilyen szinten speciális a többi típushoz képest, hiszen olyan módszerek is idetartoznak, amelyek nem csak digitális szövegfájl esetén működnek, hanem analóg formában is. A módszerek egy része azon a megfontoláson alapul, hogy elsősorban az emberi szem elől szeretnénk elrejteni az információt, nem pedig a szteganalitikai szoftverek elől. Rengeteg ilyen eljárás létezik, például az alábbiak:

- Ilyen módszer lehet az, ha az egyes sorokat (vagy szavakat, esetleg betűket) elmozdítjuk egymáshoz képest. Ez egy külső szemlélő számára nem feltűnő, ám alkalmas módszer az adatrejtésre, és analóg formában is működik.
- Szintén lehetséges megoldás lehet még üres jelek bevitele (pl. szóköz, tabulátor).
- Bizonyos karakterek lecserélésével is elrejthetünk információt (pl. CR/LF).
- Különböző szintaktikus és szemantikus módszerek is léteznek. Szintaktikus módszer például az, ha a vesszők rendhagyó használatának segítségével rejtjük el az elrejtendő információt, míg a szemantikus esetben a nyelvtani szabályokat használjuk ki az adatrejtés során (pl. különböző mondatszerkezeteket használunk).
- Végül, de nem utolsó sorban az is egy lehetséges adatrejtési módszer, amikor az amúgy ártalmatlannak tűnő nyílt szöveg egy jól definiált szabályok szerinti értelmezés szerint teljesen más jelentéssel bír (pl. szavak kezdőbetűit összeolvasva kapjuk meg az elrejtett szöveget).

Egy szöveges adatokba történő adatrejtésre használható program például a SNOW (Steganographic Nature of Whitespace) program, amely ASCII szövegben helyez el a sorok végére szóközöket, valamint tabulátorokat, így rejtve el információt. A program lehetőséget biztosít arra is, hogy az elrejtendő adatot egy megadott kulccsal (jelszóval) titkosítsuk.

A szövegbe történő adatrejtés sem a kapacitás, sem az észrevétlenség szempontjából nem a legpraktikusabb választás.

4. Adatrejtés tömörítetlen és veszteségmentesen tömörített képekben

A szteganográfia egyik leggyakoribb formája a képekbe történő adatrejtés. Ennek egyik fajtája az, amikor tömörítetlen vagy veszteségmentesen tömörített képbe szeretnénk információt elrejteni. Ilyenkor könnyebb dolgunk van, hiszen az elrejtett adatnak nem kell ellenállnia a veszteséges tömörítés folyamatának, amely során akár el is veszthetnénk a rejtett információt. Jellemző fájltípusok tömörítetlen képek esetén például a BMP és a TIFF típusok. Az alábbiakban néhány ilyen eljárás kerül ismertetésre.

4.1 Adatrejtés az LSB értékekben

Az egyik lehetséges eljárás során a kép képpontjai színének LSB (Least Significant Bit – legjelentéktelenebb bit) értékeibe rejtjük el a megfelelő adatot. RGB kódolás esetén ez három bit használatára ad lehetőséget pixelenként. A legegyszerűbb módszer például az, amikor az LSB értékeket az elrejtendő adat megfelelő bitjeinek értékére állítjuk. Az eljárásnál mindenképpen figyelni kell arra is, hogy például az összes LSB érték megváltoztatása felismerhetővé teheti az eljárást, így érdemes zajszerűen elhelyezni az információt.

4.2 Adaptív adatrejtés

Az imént említett ok miatt sajnos nem használhatjuk ki az összes képpontot az adatrejtés során. Nem adaptív adatrejtés esetén a rejtéshez használt képpontok kijelölése függetlenül történik a kép tartalmától, azonban ezáltal könnyebben felismerhető lesz az adatrejtés ténye, mivel feltűnően eltérő képpontok kerülhetnek egymás mellé. Ennél jobb megoldást biztosít az adaptív adatrejtés, amikor úgy jelöljük ki az adatrejtéshez használt képpontokat, hogy figyelembe vesszük az egyes képpontok környezetét, a kép dinamikáját. Ökölszabályként mondhatjuk, hogy célszerű az egyszínű tartományokat kerülni, és nagyobb szórású helyeket választani. Általánosságban azonban nagyon nehéz feladatnak bizonyul a megfelelő képpontok kiválasztása – ráadásul a dekódolásnál is szükség van rá, hogy könnyen meghatározhatóak legyenek azok a képpontok, amelyekben elrejtett információ található.

Egy lehetséges egyszerű megoldás, amikor a képet felosztjuk kisebb területekre, majd ezek közül a sztego kulcs által kijelölt területekre ellenőrizzük, hogy jó vagy rossz blokk – jó blokk az, amely valószínűleg jól használható adatrejtésre, míg rossz blokk esetén a benne megtalálható képpontok egymáshoz való viszonya alapján megállapíthatjuk, hogy nem lenne szerencsés adatot elrejteni a blokkban. Ha a blokkot jónak ítéltük meg, akkor adatot rejtünk bele, különben nem. Ahhoz, hogy a dekódoló oldalon is megállapítható legyen, hogy hol található rejtett információ, ki kell bővíteni az eljárást még annyival, hogy egy blokkba történő adatrejtés után meg kell vizsgálni, hogy a rejtett adattal együtt is még jó blokknak számít-e. Ha az adatrejtés „elrontotta”, akkor a következő blokkban megismételjük az adatrejtést, és így tovább. Ezáltal a dekódoló oldalon könnyen megállapítható, hogy hol található rejtett információ – a sztego kulcs segítségével meghatározható a szóba jövő blokkok halmaza, majd ezek közül csak a jó blokkok alapján végezzük el a helyreállítást.

4.3 Palettás képek

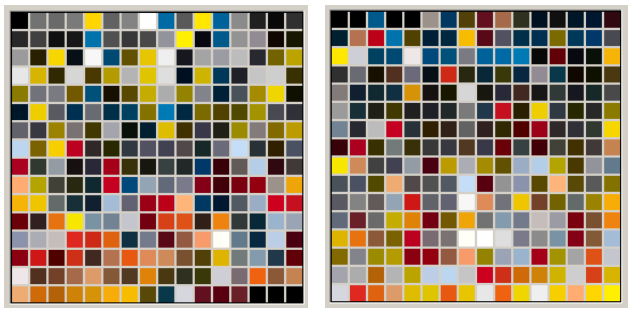
Speciális eljárás alkalmazható az úgynevezett palettás képeknél – ilyenek lehetnek például a BMP és GIF fájlformátumú képek. Palettás képek esetén a kép egyes pixeleiben csak egy hivatkozás található a paletta egy elemére (pl. a sorszámára), és a megfelelő palettaelem tartalmazza az adott képpont esetében használni kívánt szín RGB kódját. A paletta tehát tulajdonképpen az összes használható szín színkódját tartalmazó indexelt lista. A BMP fájltípus például a legfeljebb 8 bites színmélységű képek tárolása esetén használ csak palettát (8 bites színmélység esetén 256 színből áll a palettánk). Amennyiben több mint 8 bit áll rendelkezésre egy képpont esetében a színe eltárolásához, akkor már minden egyes képpontra tárolja a színkódot, nem használ palettát. A GIF fájlformátum veszteségmentes tömörítést használ, és legfeljebb 256 színt tartalmazó képet képes tárolni – a használt színek színkódjait szintén palettában tárolja.

4.4 Adatrejtés palettás képekben

A palettába történő adatrejtés legfontosabb jellemzője, hogy az elrejthető adat mennyisége nem függ a kép méretétől, azt egyedül a paletta mérete határozza meg. Ebből következően a palettás adatrejtés kapacitása erősen limitált. Ilyen képek esetében nem alkalmazhatóak közvetlenül azok a módszerek, amelyek a képek egyes képpontjaiba rejtik az adatot. Például a korábban már említett LSB kódolást sem használhatjuk egy az egyben, hiszen az egyes képpontokban most indexek helyezkednek el, melyeknek a legjelentéktelenebb bitjét megváltoztatva valószínű, hogy teljesen más színt kapunk – olyat, amely egyáltalán nem hasonlít az eredeti színre.

Az egyik legalapvetőbb módszer palettás képek esetén, ha a palettában megtalálható színeket nem módosítjuk – ezáltal az adatot rejtő kép teljesen pontosan meg fog egyezni az eredeti képpel –, hanem a palettát permutáljuk, azaz felcseréljük az egyes színek sorszámait. Az eljárást detektálhatóvá teheti azonban a gyanúsan kevert paletta. Például az ingyenesen letölthető Gifshuffle program az imént ismertetett palettapermutáló eljárást használja GIF formátumú fájlok esetén.

Az alábbiakban egy 256 színből álló paletta eredeti verziója (bal oldalt), valamint permutáció utáni verziója látható (jobb oldalt).



3. ábra: Egy 256 színű paletta eredeti és permutáció utáni állapota

Másik lehetséges megoldást jelenthet a korábban már ismertetett LSB kódolás használata, azonban a korábbiaktól eltérő módon itt a paletta színeinek LSB értékeit használjuk az adat elrejtésére. Ekkor színenként három hasznos bit áll rendelkezésünkre, amely egy 256 színből álló palettánál 768 bitet, azaz 96 bájtot jelent.

A paletta színeinek csökkentésének segítségével is elrejthetünk adatot palettás képekben. Egyik ilyen megoldás, hogy csökkentjük a színmélységet, majd a rejtett adatnak megfelelően terjesztjük ki a színeket. Például egy 256 színből álló palettát redukálhatunk 32 színre, majd az elrejtendő információtól függően bővíthetjük úgy 256 színre úgy, hogy több egymáshoz közeli színünk lesz. Ebben az esetben három hasznos bitünk lesz minden egyes színkódban, az információt pedig az egymáshoz közeli színek hordozzák. Az egyes képpontokban az egymáshoz közel álló színek közül bármelyiket használhatjuk, így az eljárás nem is olyan feltűnő. Az is lehetséges módszer, hogy a színmélység csökkentése után valóban csak annyi színt használunk, amennyire redukáltuk a színek számát, majd a paletta felszabadult helyeire tesszük az információt. A módszer előnye az előző megoldáshoz képest, hogy jóval több hely áll rendelkezésünkre az adat elrejtésére, elég feltűnő jelenség azonban, hogy nincsenek közeli színek, így könnyebben detektálható az eljárás.

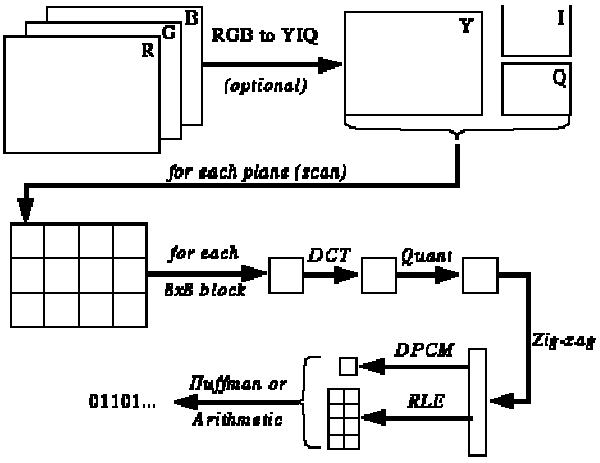
5. Adatrejtés veszteségesen tömörített képekben

Veszteségesen tömörített képek esetén nem alkalmazhatóak a korábban említett módszerek, hiszen ekkor könnyen előfordulhat, hogy a tömörítés során elvesztjük az elrejtett információt. Ebből adódóan teljesen más, új adatrejtő eljárásokra van szükség ilyen képek esetében. Az egyik legelterjedtebb veszteséges tömörítést használó képformátum a JPEG, így a továbbiakban elsősorban erről lesz szó. A JPEG-ben történő adatrejtési lehetőségek megértéséhez azonban feltétlenül szükséges a JPEG kódolási, tömörítési eljárásának ismerete.

5.1 JPEG kódolás menete

A JPEG kódolás nagyon röviden összefoglalva a következő fázisokból áll:

- A kép RGB összetevői alapján luminancia (Y) és krominancia (IQ vagy UV) összetevők meghatározása.
- A kép felosztása 8×8-as blokkokra.
- Az egyes blokkok diszkrét koszinusz transzformációja (DCT), melynek eredményeképp DC és AC komponenseket kapunk.
- A transzformált blokkok kvantálása – ebben a fázisban történik az adatvesztés.
- A kvantált blokkok veszteségmentes tömörítése.



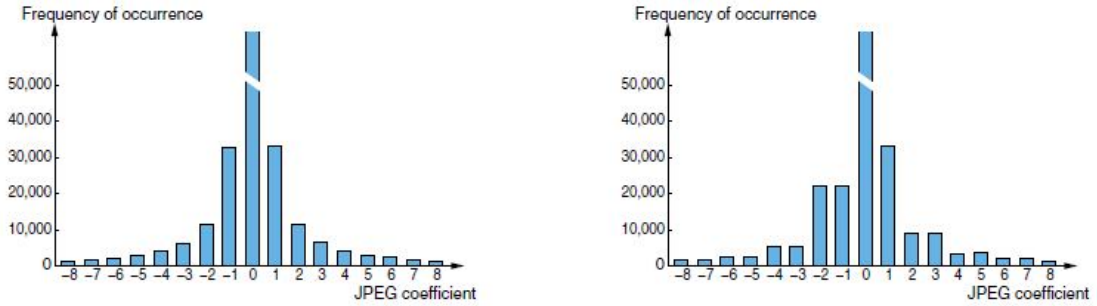
4. ábra: JPEG kódolás folyamata

5.2 Adatrejtés JPEG formátumú képekben

JPEG képekbe történő adatrejtésre nagyon sok algoritmus létezik, ezek közül kerül pár megemlítésre az alábbiakban.

Adatot rejthetünk el egy JPEG képben például a kódolás kvantálási fázisában úgy, hogy kihasználjuk, hogy kerekítést hajtunk végre a kvantálás során. Amennyiben a kerekítendő érték törtrésze 0,5-höz közeli, úgy a kerekítést az elrejtendő adat függvényében hajtjuk végre. Ahhoz, hogy az eljárás dekódolható legyen, egyrészt szükséges az elrejtendő információ hibajavító kódolása, azaz redundánsan kell elrejteni az adatokat, másrészt pedig a visszafejtés során szükség van az eredeti képre, mivel össze kell hasonlítani az adatrejtési helyeken a kerekítéseket az eredeti és a sztego médium esetében.

Egy használható algoritmus ezen kívül még a JSteg algoritmus, amely a kép DCT együtthatóiba képes adatot rejteni úgy, hogy az AC együtthatók LSB értékét használja. Az eljárás hibája, hogy az AC együtthatók frekvenciadiagramon történő ábrázolásának segítségével könnyen detektálható, mivel a diagram szimmetriája megbomlik. Az alábbiakban látható az említett frekvenciadiagram az eredeti kép (bal oldalt) és a sztego médium esetében (jobb oldalt).



5. ábra: JSteg algoritmus esetén az eredeti és a sztego médium AC együtthatóinak frekvenciadiagramja

Hasonló elveken működő algoritmusok az F3, F4 és F5 algoritmusok, amelyek viszont már nem bontják meg a frekvenciadiagram szimmetriáját, így a JSteg esetében működő felismerés itt nem működik. Az F5 algoritmus újítása még a JSteg algoritmushoz képest például az is, hogy az elrejtett információt nem a kép „elejétől” kezdve rejtjük el folyamatosan, hanem a képet először álvéletlen permutáljuk, a permutált képben rejtjük el folyamatosan az adatot, majd visszaállítjuk az eredeti sorrendet – ezáltal az elrejtett információ szétszóródik a képben. Az F5 algoritmus ugyan a JStegnél használt eljárással nem ismerhető fel, viszont felismerhető például olyan módszerrel, amely az eredeti kép AC együtthatóinak becslésén alapul – sőt, nem csak az adatrejtés ténye fedhető így fel, hanem a rejtett üzenet hossza is.

Az OutGuess program algoritmusa az F5 algoritmuson alapszik, azonban kiküszöböli annak a hibáját, miszerint a frekvenciadiagram segítségével detektálható az adatrejtés ténye. Az OutGuess annyival egészíti ki az F5 algoritmust, hogy az F5 során fel nem használt AC együtthatók segítségével visszaállítja az AC frekvenciadiagramot eredeti állapotára. Ezáltal a frekvenciadiagram alapján valóban nem fedhető fel az információrejtés ténye, azonban az OutGuess megváltoztatja a képtartománybeli blokkosodás mértékét, melyet kihasználva az OutGuess-szel történő adatrejtés felismerésére is létezik viszonylag könnyen implementálható eljárás. Az OutGuess is szabad felhasználású szoftver, így ingyenesen letölthető.

6. Adatrejtés hangfájlokban

A hangfájlokba történő adatrejtés valamivel kevésbé elterjedt formája a szteganográfiának, mint a képekbe történő adatrejtés. A hangfájlok esetén is mindenképpen meg kell különböztetni a tömörítetlen hangot (pl. CD felvétel) a veszteségesen tömörítettől (pl. MP3), hiszen itt is teljesen más eljárások alkalmazhatóak a két esetben.

Egyszerű adatrejtési módszerként használható tömörítetlen hangfájl esetében a képeknél már említett LSB értékek módosítását használó módszer. Az eljárás hátránya, hogy az emberi fül számára hallható lehet a bevitt zaj ellentétben azzal, hogy a képek esetében nem okozott szabad szemmel látható változást az eljárás. Az eljárást tovább lehet azonban úgy fejleszteni, hogy a sztego kulcs alapján különböző LSB csoportokat választunk, amelyeknek a paritása fogja megadni az elrejtett információt. Így az adott tartományokon belül szabadon választhatunk, hogy melyik bitet billentjük át ahhoz, hogy a megfelelő paritást kapjuk. Ennek köszönhetően adaptívan tudjuk elrejteni a megfelelő adatot a hangfájlba.

Az adatrejtés alapvető célja hangfájlok esetében elsősorban az, hogy az emberi fül számára ne legyen érzékelhető az adatrejtés ténye. Az emberi fül egyik sajátossága, hogy tipikusan 20-20000 Hz közötti tartományba eső hangokat hallja meg csupán. Másik megemlítendő tulajdonság, hogy két közeli frekvencián lévő jel közül az erősebb jel teljesen elnyomja a gyengébbet. Ezen tulajdonságokat kihasználva is rejthetünk adatot hangfájlokba, amelynek köszönhetően az emberi fül számára nem lesz detektálható az adatrejtés.

Lehetséges módszerként felmerül még olyan eljárás is, amely a hangfájlokban jelenlévő visszhangokba rejt információt úgy, hogy plusz hangokat ad a visszhanghoz. Az eljárás előnye, hogy nem csak hogy nem rontja a hangzás minőségét, hanem akár még úgy is érezhetjük, mintha javulna a hangzás.

A Steghide nevű program segítségével adatot rejthetünk hangfájlokba, így például WAV formátumú hanganyagba. A program ezen felül képekbe történő adatrejtésre is lehetőséget biztosít, és szintén ingyenesen letölthető.

7. Adatrejtés videófájlokban

A videófájlok esetén használhatóak a korábban már ismertetett képekbe és hangfájlba adatot rejtő módszerek, hiszen egy videófájl tulajdonképpen sok képfájl és hang együttese. Alapvetően két különböző kódolási technikát különböztethetünk meg videófájlok esetén:

- *Intra-frame kódolás*: az adott képkockát tömörítjük a többitől függetlenül, nincs időbeli tömörítés
- *Inter-frame kódolás:* felhasználjuk, hogy az egymást követő képek hasonlóak, így egy referenciaképhez történő mozgásbecslés alapján kódoljuk a képet

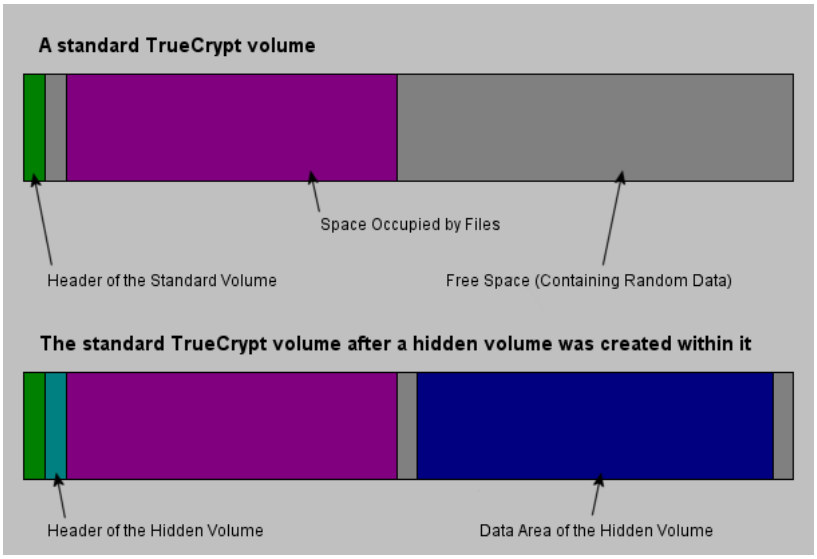
Vannak olyan videótömörítési algoritmusok, amelyek minden egyes képre intra-frame kódolást alkalmaznak, gyakoribbak azonban azok az eljárások, amelyek néhány referenciaképet intra-frame kódolással kódolnak, majd a többi képet ezekhez képest inter-frame kódolás alkalmazásával

tömörítik. Mindkét módszer közös tulajdonsága azonban, hogy tartalmaz olyan képkockákat, amelyeket csak önmagukban tömörítünk (előbbi esetben mindegyik képkocka ilyen, utóbbi esetben csak a referenciaképek). Ezen képkockák tömörítésére általában diszkrét koszinusz transzformációt szokás használni, így például a JPEG tömörítésnél ismertetett, illetve azokhoz hasonló módszerek használhatók ezekbe a képekbe történő adatrejtéshez. A korábban már említett hanganyagokba történő adatrejtési módszerek pedig gyakorlatilag egy az egyben felhasználhatóak videóanyagok hangja esetén is.

8. Adatrejtés futtatható állományokban és merevlemezen

Léteznek olyan szteganográfiai adatrejtő megoldások, amelyek segítségével futtatható állományokban is el tudunk rejteni információt. Ezek a megoldások főként azon az elven alapulnak, hogy egy feladat gyakran sokféleképp is elvégezhető. Arra kell csak az adatrejtés során figyelni, hogy a program funkcionalitása megmaradjon, illetve a különböző processzorarchitektúrákra is tekintettel kell lenni. Intel i386-os processzorarchitektúrán például a Hydan ingyenesen letölthető program segítségével rejthetünk el adatokat futtatható fájlokba, amely az i386 utasításkészletében rejlő redundanciát használja ki.

A TrueCrypt alapvetően egy merevlemezek titkosítására használható alkalmazás, emellett azonban adatok elrejtésre is használható, így segítségével megvalósítható a merevlemezen történő adatrejtés. Az eljárás lényege, hogy egy rejtett TrueCrypt kötetet hozunk létre egy másik, titkosított TrueCrypt köteten belül. Mivel a TrueCrypt a titkosított kötetek végén az üres helyet véletlen adatokkal tölti fel amúgy is, így egy külső megfigyelő nem tudja megállapítani, hogy rejtett tartalom helyezkedik-e el a köteten, hiszen az is véletlenszerűnek tűnik számára. A módszert az alábbiakban látható ábra szemlélteti.



6. ábra: Adatrejtés TrueCrypt segítségével egy merevlemezen

9. Szteganalízis

Szteganalízisnek a rejtett információk észlelésének eljárását nevezzük, melynek célja csupán a rejtett kommunikáció felfedése. Ez azt jelenti, hogy elsősorban már azzal is megelégszünk, ha az információ jelenlétét kimutatjuk – az már csak plusz feladatnak számít, ha az elrejtett információt is vissza szeretnénk nyerni a sztego médiumból.

A szteganalízis bizonyos szempontból hasonlít a kriptanalízishez, hiszen itt is az előforduló gyengepontok feltárására koncentrálunk. Emellett azonban a szteganalízis mutat egy nagyon fontos különbséget a kriptanalízishez képest. Ez a különbség abban rejlik, hogy kriptográfia esetében rendelkezésünkre áll egyértelműen a titkosított információ, amelyet meg kell fejtenünk, szteganográfia esetében azonban már az is komoly feladatot jelenthet, hogy meg tudjuk állapítani a különböző médiumokból, hogy melyikbe rejtettek adatot.

A bemutatott szteganográfiai eljárások nagy része olyan, hogy az adatrejtés során sztego kulcsot használunk, azaz kulcs felhasználásával ágyazzuk be az információt a hordozó médiumba. Ebből adódóan ezen eljárások esetében, még ha a detektálás esetleg könnyű feladat is, az üzenet visszafejtése már jóval bonyolultabb feladatot jelent.

A különböző szteganalitikai szoftverek általában különböző statisztikai, valamint heurisztikus módszereket használnak az adatrejtés detektálására. A StegDetect egy nyilvános forráskódú szoftver, amely például a korábban említett JSteg, OutGuess és F5 algoritmusokat is képes detektálni. Emellett azonban még rengeteg szteganalitikai eszköz létezik, amelyeknek viszont jelentős része fizetős szoftver.

10. Összefoglalás

Összességében elmondható, hogy a szteganográfia ugyan egy nagyon régi, már az ókorban is ismert tudományág, ám napjainkban mégis virágkorát éli. Ez legfőképp annak köszönhető, hogy a digitális rendszerekben rendkívül sokféle médium áll rendelkezésünkre, melyekben adatokat rejthetünk el. Láthattuk, hogy szöveges anyagokba, képfájlokba, hang- és videóanyagokba, de még akár futtatható állományokba is minden további nélkül elrejthetünk információt. A hordozó médium típusától függően azonban rendkívül sok eljárás létezik, hiszen minden médiumban más technikával tudjuk az adatrejtést végrehajtani. Láthattuk azt is, hogy nem csak a hordozó médium fájltípusától függ, hogy milyen algoritmust használhatunk, hanem a fájl pontos formátumától is – nagyon lényeges információ például az is, hogy tömörített vagy tömörítetlen médiumba szeretnénk adatot rejteni.

A bemutatott eljárások jelentős részéhez létezik már megfelelő szoftver. A programok nagy része ingyenesen letölthető, és használatuk általában annyira egyszerű, hogy egy átlagos felhasználó is könnyedén tud adatot rejteni valamilyen médiumba. Emellett az ismertetett módszerek nagy részének detektálására is már léteznek megfelelő programok, melyek segítségével felismerhető, hogy egy adott médiumban van-e elrejtve valamilyen információ. Ebből adódóan gyakorlatilag folyamatosan szükség van újabb és újabb szteganográfiai algoritmusok megalkotására.

11. Hivatkozásjegyzék

[1] FÖLDES ÁDÁM MÁTÉ: *A szteganográfia és annak relevanciája a privátszféra védelmében,* Hacktivity, 2008.
http://hacktivity.hu/portal/archivum/folia/2008/foldes_mate_adam-a_szteganografia.pdf

[2] FEHÉR GÁBOR: *„Médiabiztonság”* előadás, BME-TMIT, 2010.
http://qosip.tmit.bme.hu/twiki/pub/Main/MediaBiztonsag/02-Szteganografia.pdf
http://qosip.tmit.bme.hu/twiki/pub/Main/MediaBiztonsag/03-Adatrejtes_kepekben.pdf
http://qosip.tmit.bme.hu/twiki/pub/Main/MediaBiztonsag/06-Adatrejtes_hanganyagokban.pdf

[3] UNICSOVICS GYÖRGY: *Bevezetés a szteganográfiába*, HTE előadás, 2007.
http://www.eoq.hu/akt2/inf70326.pdf

[4] UNICSOVICS GYÖRGY: *Bevezetés a szteganalízisbe*, HTE előadás, 2007.
http://www.eoq.hu/akt2/inf71119.pdf

[5] ANDREAS WESTFELD: *F5 – A Steganographic Algorithm: High Capacity Despite Better Steganalysis*, Technische Universität Dresden, 2001.
http://os.inf.tu-dresden.de/~westfeld/publikationen/f5.pdf

[6] WIKIPÉDIA – A SZABAD ENCIKLOPÉDIA: *Szteganográfia*
http://hu.wikipedia.org/wiki/Szteganogr%C3%A1fia

[7] WIKIPEDIA – THE FREE ENCYCLOPEDIA: *Steganography*
http://en.wikipedia.org/wiki/Steganography

12. Programjegyzék

[1] SNOW (STEGANOGRAPHIC NATURE OF WHITESPACE)
http://www.darkside.com.au/snow/

[2] GIFSHUFFLE
http://www.darkside.com.au/gifshuffle/

[3] OUTGUESS
http://www.outguess.org

[4] STEGHIDE
http://steghide.sourceforge.net/

[5] HYDAN
http://www.crazyboy.com/hydan/

[6] TRUECRYPT
http://www.truecrypt.org/