IT Security Coding Contest 2009

1 Feladatkiírás

1.1 Általános szabályok

A verseny során bármilyen segédeszközt használhattok a feladatok megoldásához.

A feladatok megoldását csak abban az esetben fogadjuk el, ha az alábbi anyagokat is megkapjuk:

- A megoldó program vagy programok használatának leírása, valamint a futtatásukhoz szükséges előfeltételek (operációs rendszer, futtató környezet (.net, jre,...), stb.)
- A megoldó program vagy programok bináris (futtatható) formában – működőképesen! (Amennyiben van valamilyen függőség, azokat a dll-eket, könyvtárakat, stb. is kérjük!)
- A megoldó program vagy programok forráskódja
- A fordításhoz szükséges információk (használt fordító megnevezése, pontos verziója, használt kapcsolók, stb.)
- A megoldás elvét leíró dokumentáció
- A csapat tagjainak nevét tartalmazó txt fájl

Minden egyes feladat megoldását (azaz a fenti anyagokat mind) külön csomagolt (zip/rar/tar.gz,…) állományban várjuk.

Az értékelés során általános irányelvként figyelembe vesszük a megoldás eredményességét, hatékonyságát, valamint megvalósítás igényességét. Részpontokat is adunk, érdemes kész megoldásokat is beadni.

1.2 Feladat kiírás – Aki keres az talál!

Adott 5 db Windows Bitmap képfájl, ezek közül legalább egyben elrejtettünk egy egyszerű, nem kulcsos szteganográfiai módszerrel egy nagyméretű (több tíz kbyte) adatot, amit ráadásul nem is titkosítottunk.

A feladat igen egyszerű: meg kell találni, hogy melyikben van elrejtve a titok, és ki is kell onnan venni azt.

A beadandó program az alábbi paraméterezéssel működjön:

stego input_file.bmp output_file

Ahol az input_file.bmp a titkot tartalmazó BMP fájl, az output_file pedig a „kibontott” titkot tartalmazza majd.

FONTOS: A feladatnak nem csupán a dekódolás a célja, hanem az is, hogy meg tudjátok magyarázni a rejtés elvét, illetve meg is tudjátok mutatni, hogy miért nincs a többi fájlban adat.

Egyszóval várnánk egy napló-szerű leírást arról, hogy mit/miért/hogyan csináltatok.

2 Megoldás

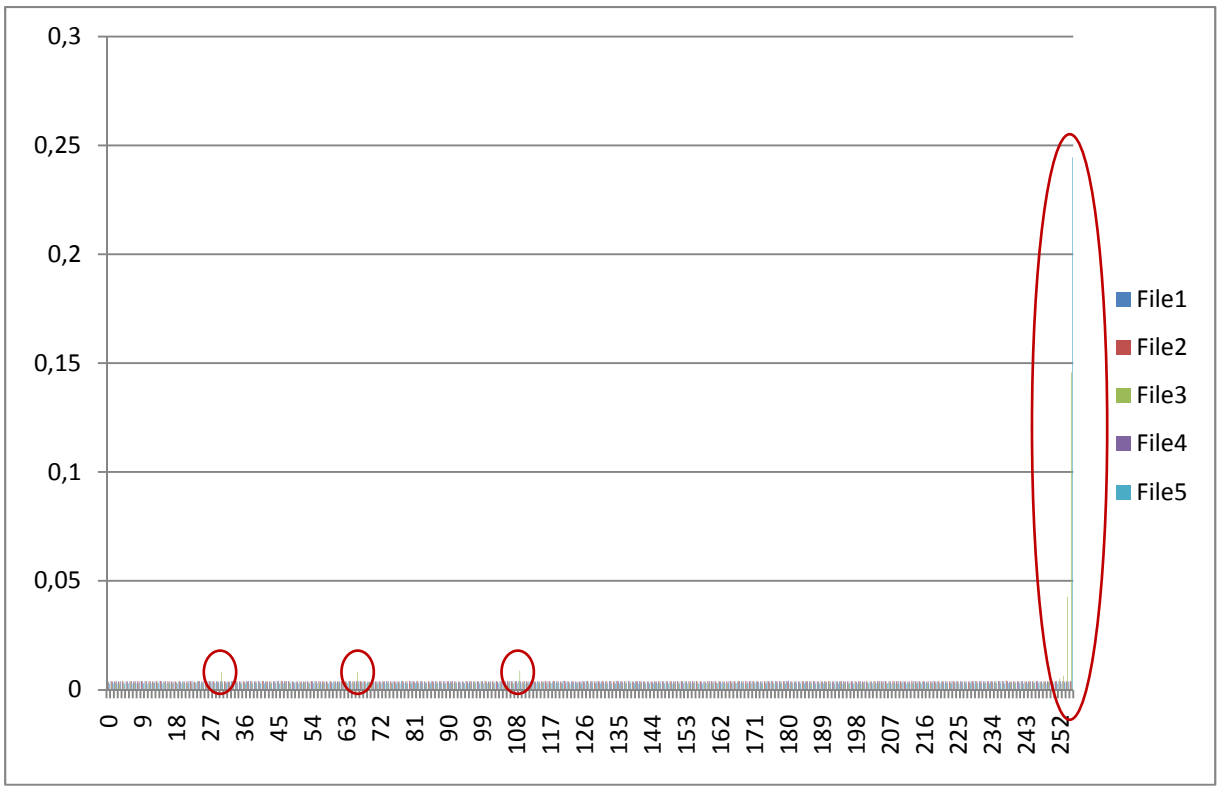
A megoldáshoz annyit kell tudni a 32 bites BMP fájlokról, hogy először egy 54 byte méretű fejléc szerepel a fájl elején, utána pedig a képpontok következnek, amelyek 4 byte méretűek (32 bit), és BGRA felépítésűek: kék, zöld, piros és alpha csatornából állnak. Az alpha csatorna felelős a képpont áttetszőségéért, viszont mivel ezt a programok zöme[1] nem veszi ezt figyelembe, ez alkalmassá válik képpontonként 1 byte információ elrejtésére.

Emellett még LSB módszerrel is lehetséges lenne a rejtés, ám két okot is találhatunk, hogy miért inkább a másik irányba induljunk el: egyrészt valamennyi képfájl alpha byte-jai tele vannak írva (a programok rendszerint 0 értékkel írják tele az alpha csatornát), valamint nagyobb méretű adatok rejtésére ez a módszer alkalmasabb. Sokféleképp vizsgálhatjuk az alpha csatornát, most csak egy egyszerű megoldást teszünk közzé.

Tegyük fel, hogy kiolvastuk az értékeket belőle minden egyes képnél, és megvizsgáljuk, hogy az egyes értékek milyen sűrűn fordulnak elő. Az egyes képfájlok alpha byte-jainak a várható értéke az alábbi:

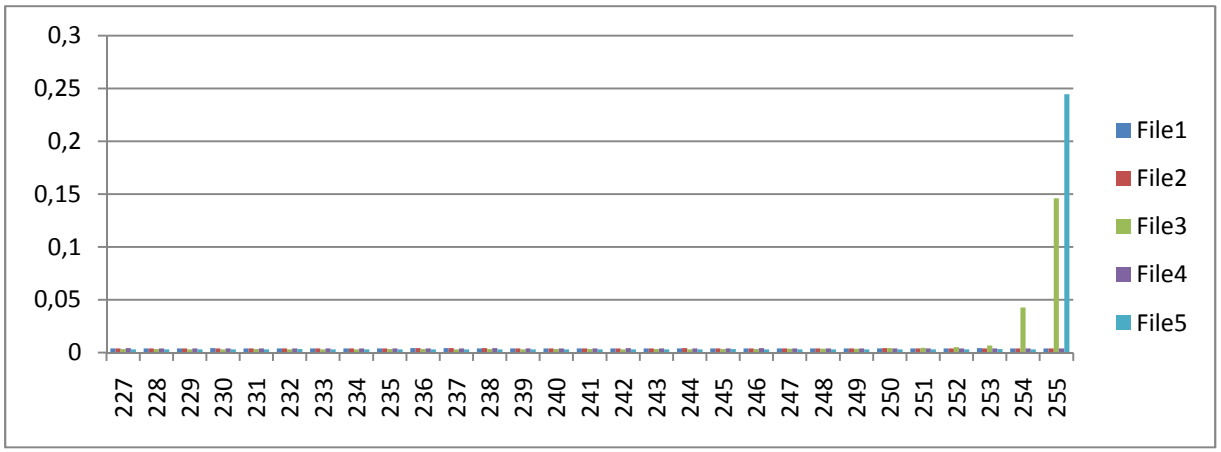
1. kép: 127,370 byte
2. kép: 127,569 byte
3. kép: 150,529 byte
4. kép: 127,548 byte
5. kép: 158,392 byte

Ezt jól mutatja az alábbi grafikon is, ahol az látható, hogy mekkora valószínűséggel fordultak elő fájlonként az egyes byte értékek. Látható, hogy a képek zöménél a zaj jelleg dominál, de kettő kilóg a sorból.



[1] Az XnView például megmutatja: http://www.xnview.com/en/index.html

Kiemelve a jobb oldali részt (bár itt más kiugrások is vannak):



Látszik, hogy a 3. és 5. fájlokban érdemes keresgélni.

Ha ezeket a fájlok alpha csatornáit megnézzük, látható, hogy a kiugró értékek mintázatszerűen vannak belekódolva, azaz minden negyedik helyen van "gyanús" érték. Ezeket szekvenciálisan kiolvasva kinyerhető két bytefolyam. Az ötödik fájl esetén egy homogén, 255 értékű byte-okat tartalmazó fájl lesz a végeredmény, míg a harmadik fájl esetén megkapjuk a keresett fájlt, a PET Portál logóját tartalmazó BMP fájlt.

Ezzel meg is oldottuk a feladatot.