

# ونھان سازی دادہا



... ✨

چت ھازمر



پوشیده‌نگاری

و نهان‌سازی داده‌ها

نویسندگان :

میکائیل راگو

چت هازمر

برگردان: علیرضا اصغری



|                     |  |
|---------------------|--|
| سرشناسه             | : راگو، مایکل تی.  |
| عنوان و نام پدیدآور | : Raggio, Michael T  |
| مشخصات نشر          | : پوشیده‌نگاری و نهان‌سازی داده‌ها/ نویسندگان میکائیل راگو، چت هازمر؛ برگردان علیرضا اصغری.  |
| مشخصات ظاهری        | : سنندج: علیرضا اصغری، ۱۳۹۵.   |
| شابک                | : ۳۰۶ ص.   |
| وضعیت فهرست نویسی   | : 978-600-04-5453-1  |
| یادداشت             | : فیا  |
| موضوع               | : عنوان اصلی: , 2013, Data hiding : exposing concealed data in multimedia, operating systems, mobile devices and network protocols |
| موضوع               | : کامپیوترها -- ایمنی اطلاعات  |
| موضوع               | : Computer security  |
| موضوع               | : حفاظت داده‌ها  |
| موضوع               | : Data protection  |
| شناسه افزوده        | : هازمر، چت  |
| شناسه افزوده        | : Hosmer, Chet   |
| شناسه افزوده        | : اصغری، علیرضا، ۱۳۵۵ -، مترجم   |
| رده بندی کنگره      | : ۱۳۹۵ ر۲۴ / ک۹۶ / ۹ / QA۷۶  |
| رده بندی دیویی      | : ۰۰۵ / ۸  |
| شماره کتابشناسی ملی | : ۴۲۹۷۳۶۷  |

برگردان : علیرضا اصغری  
ویراستار: پرویز گلپسندی







## فهرست

|     |       |  |
|-----|-------|--|
| ۱۱  | ..... | دیباچه   |
| ۱۵  | ..... | مقدمه  |
| ۱۷  | ..... | فصل ۱ تاریخچه رمزنگاری   |
| ۱۷  | ..... | پیشگفتار   |
| ۱۹  | ..... | دانش رمزنگاری  |
| ۲۸  | ..... | پوشیده‌نگاری   |
| ۳۷  | ..... | چکیده  |
| ۳۹  | ..... | فصل ۲ چهار روش ساده‌ی پنهان‌سازی داده‌ها                               |
| ۴۰  | ..... | پنهان‌سازی داده‌ها در نرم‌افزار ورد                                    |
| ۴۸  | ..... | ابرداده‌های در فایل تصاویر   |
| ۵۲  | ..... | پنهان‌سازی داده‌ها در ابزارهای همراه                                   |
| ۵۸  | ..... | پنهان‌سازی داده‌ها در نرم‌افزارهای فشرده‌سازی فایل                     |
| ۶۴  | ..... | چکیده  |
| ۶۷  | ..... | فصل ۳ پوشیده‌نگاری   |
| ۶۹  | ..... | شیوه‌های پوشیده‌نگاری  |
| ۸۶  | ..... | تحلیل پوشیده‌نگاری   |
| ۹۹  | ..... | چیکده  |
| ۱۰۱ | ..... | فصل ۴ پنهان‌سازی داده‌ها در فایل‌های چندرسانه‌ای                       |
| ۱۰۱ | ..... | مروری بر چندرسانه‌ای   |
| ۱۰۲ | ..... | پنهان‌سازی داده‌ها در صوت دیجیتال                                      |
| ۱۱۳ | ..... | پنهان‌سازی داده‌ها در فایل‌های ویدئویی دیجیتال                         |
| ۱۱۴ | ..... | چکیده  |
| ۱۱۵ | ..... | فصل ۵ پنهان‌سازی داده‌ها در ابزارهای همراه اندرویدی                    |
|     |       | پنهان‌سازی داده‌ها در ابزارهای اندرویدی به وسیله‌ی نرم‌افزار Img Hid و |

|   |     |
|---|-----|
| APP آشکارسازی آن  | ۱۱۵ |
| چکیده   | ۱۴۰ |
| <b>فصل ۶ پنهان سازی داده در سیستم عامل اپل</b>                | ۱۴۱ |
| نرم افزارهای پنهان سازی داده ها در ابزارهای همراه             | ۱۴۱ |
| چکیده   | ۱۷۵ |
| <b>فصل ۷ پنهان سازی داده ها در سیستم عامل ویندوز و لینوکس</b> | ۱۷۷ |
| پنهان سازی داده ها در سیستم عامل ویندوز                       | ۱۷۹ |
| پنهان سازی داده ها در لینوکس                                  | ۱۹۷ |
| <b>فصل ۸ پنهان سازی داده ها به شکل مجازی</b>                  | ۲۱۹ |
| پنهان سازی در محیط مجازی                                      | ۲۲۰ |
| مرووری بر محیط های مجازی                                      | ۲۲۴ |
| چکیده   | ۲۳۳ |
| <b>فصل ۹ پنهان سازی داده ها در پروتکل های شبکه</b>            | ۲۳۵ |
| پنهان سازی داده ها در VOIP                                    | ۲۳۸ |
| کشف پنهان سازی داده ها در پروتکل شبکه                         | ۲۴۶ |
| چکیده   | ۲۴۷ |
| <b>فصل ۱۰ تحقیقات قضایی و راه گریز از آن</b>                  | ۲۴۹ |
| پیگرد قضایی   | ۲۵۶ |
| جستجوی فایل ها و پوشه های پنهان                               | ۲۷۱ |
| سیستم تشخیص مهاجم در شبکه ها                                  | ۲۷۲ |
| چکیده   | ۲۷۵ |
| <b>فصل ۱۱ بازرسی قضایی</b>                                    | ۲۷۷ |
| فناوری شناسایی داده های پنهان شده در شبکه                     | ۲۸۴ |
| فناوری های نوین برای شناسایی پنهان سازی داده                  | ۲۹۰ |
| چکیده   | ۲۹۲ |
| <b>فصل ۱۲ نگاهی به گذشته و نیم نگاهی به آینده</b>             | ۲۹۷ |
| فناوری بی سیم- یافته های جدید                                 | ۲۹۷ |
| تهدیدات ترکیبی فعلی و آتی                                     | ۳۰۲ |
| چکیده   | ۳۰۳ |

پیشکش به همسر گرامیم، مادر دلبندانم تینا و ایلینا، به پاس زحماتی  
که در این راه کشید و مرا یاری نمود.



## دیباچه

سپیده دم کشورگشایی و شکل‌گیری امپراتوری‌ها و چیرگی بر سرزمین‌های دیگر، در پرتوی برتری نظامی و با تکیه بر توانایی ارتش‌ها در گردآوری اطلاعات و فرستادن پیام‌ها و دستورهای نظامی بدون آگاه نمودن دشمن از آنها آغاز و سبب پایه‌ریزی رمز و رمزنگاری در تاریخ شده و به این دلیل است که ارتباطات سری، تاریخچه‌ای جالب در تمدن‌ها، جنگ‌ها و فرهنگ‌های گوناگون دارد. امروزه بسیاری از روش‌های برجسته‌ی رمزنگاری و ارتباطات سری دوران گذشته، به شیوه‌ای ظریف به دنیای دیجیتال امروز نیز راه یافته است و با پیدایش کامپیوتر، پیچیدگی رمزنگاری به طور چشمگیری افزایش یافته و این افزایش قدرت پردازش کامپیوترها سبب ترکیب دو شیوه‌ی تغییر محل و تغییر شکل حروف در یک روش رمزنگاری به صورت همزمان شده است. رمزنگاری کلاً بر دو گونه است: ۱- نوشتن پنهان: به روشی از نگارش داده‌ها گفته می‌شود که برای چشم غیرمسلح قابل مشاهده بوده ولی بدون تجزیه و تحلیل بی‌معنی باشد. ۲- استتار: نوشتن در لفافه و یا نوشتن نامرئی و نوشتن به شکلی است که توسط چشم غیرمسلح قابل مشاهده نباشد. نخستین نمونه‌ی استفاده از جوهر نامرئی به قرن اول میلادی بر می‌گردد، شاید شناخته‌شده‌ترین جوهر نامرئی، آب لیمو است که نوشته‌اش پس از خشک شدن برای چشم غیرمسلح نامرئی بوده، اما در مجاورت منبع گرما، متن به آرامی آشکار می‌گردد.

پوشیده‌نگاری شکلی از استتار داده‌هاست که تلاش می‌کند داده‌های موجود در فایل‌های حامل دست نخورده به نظر برسند. پوشیده‌نگاری به دو روش انجام می‌گیرد: ۱- درج: داده‌های موجود در فایل تغییر نمی‌کند، بلکه داده‌های دیگری به فایل افزوده می‌شود. ۲- جایگزین: داده‌های موجود در فایل بدون این که داده‌ای به آن‌ها اضافه شود، تغییر می‌کنند. در هر دو حالت اندازه‌ی فایل تغییر می‌کند، اما در روش درج، داده‌هایی به اطلاعات اصلی افزوده می‌شود، درحالی‌که در روش تغییر، داده‌های موجود فایل عوض می‌شود. پوشیده‌نگاری در انواع فایل‌ها همچون عکس، فیلم، متن، فایل اجرایی و... حتی در پروتکل‌های شبکه پیاده‌سازی می‌شود. برای پنهان‌سازی داده‌ها در فایل صوتی، نخست باید سیستم شنیداری انسان را بشناسیم. سیستم شنیداری انسان با داشتن ویژگی حس شنیداری دقیق و حساس، امکان پنهان‌سازی داده‌ها در فایل‌های صوتی را مشکل می‌نماید. اما اگرچه سیستم شنیداری انسان طیف گسترده‌ای دارد اما دامنه‌ی تشخیصی محدودی داشته و قادر به درک حالت مطلق نیست و تنها حالت نسبی را تشخیص می‌دهد. این محدودیت پایه و اساس روش‌های پنهان‌سازی به‌کاررفته برای فریب حس شنیداری انسان را پی‌ریزی می‌نماید. ویژگی پنهان‌سازی داده‌ها در فایل‌های صوتی در قامت فایل حامل، به لحاظ اندازه‌ی بزرگ، گنجایش مقادیر چشم‌گیری داده‌ی اضافی دارد (کل کارهای شکسپیر را می‌توان تنها در یک آواز ۸ دقیقه‌ای پنهان

نمود). تغییر کم‌ارزش‌ترین بیت در فایل‌های صوتی، شیوه‌ی پنهان‌سازی عالی و غیرقابل تشخیصی را ارائه می‌کند. به عبارت دیگر، در گوش دادن به صوت اصلی و صوت تغییر یافته، حتی حساس‌ترین گوش‌ها هم توانایی تشخیص تفاوت‌های بین این دو را ندارند. پنهان‌سازی و استتار داده‌ها در داده‌های ویدیویی دیجیتال نیز، پتانسیل بسیار بالایی در کسوت کانال پوششی دارد که بیشتر به خاطر اندازه‌ی بزرگ‌تر این فایل‌ها و تعداد زیاد فایل‌های ویدیویی موجود و کاربرد گستردی است که امروزه در اینترنت و در فضای Cloud دارند. پنهان‌سازی داده‌ها در دل عکس دیگری یکی دیگر از روش‌های پنهان‌سازی داده‌هاست به گونه‌ای که مشاهده‌ی همزمان فایل اصلی و فایل دیگری که عکس را در دل خود پنهان کرده، هیچ تأثیری روی بیننده‌ی تیزبین ندارد. یکی دیگر از حوزه‌های جالب پنهان‌سازی داده‌ها، فناوری VOIP است. زیرا به شکل گسترده‌ای کاربردی بوده و از آن استفاده می‌شود. VOIP تعداد زیادی بسته‌ی کوچک تولید می‌کند که برای پنهان کردن بخش‌های کوچک از پیام طولانی، بسیار ایده‌آل و مناسبی است. با توجه به گوناگونی گسترده‌ی نوع بسته‌ها، انواع کدها و روش‌های گوناگون کدگذاری، این فناوری را به پوشش مناسبی بدل کرده که کشف آن مثل پیدا کردن سوزن در انبار کاه است. روش جاسازی اطلاعات پنهان در پروتکل‌های شبکه با وجود میلیاردها پیام درخواست ارسال صفحات وب، ارتباطات VOIP، درخواست پخش موسیقی و ویدیو که تولید چند ده میلیارد بسته در روز در اینترنت می‌کنند، به سادگی امکان پنهان‌سازی پیام‌های کوچک و حتی مقدار زیاد اطلاعات را می‌دهند.

هرچه از قرن ۲۱ سپری می‌شود، شاهد گسترش روزافزون دستگاه‌های هوشمند همراه و ارتباطات بی‌سیم هستیم و هر روزه چشم‌به‌راه پیدایش اشکال و کاربردهای نوینی از پنهان‌سازی داده‌ها هستیم. اکنون به دور از چشم کاربر عادی، عکس گرفته شده با تلفن هوشمند، دربرگیرنده‌ی مختصات GPS محل عکس برداری، نوع دوربین، شماره سریال تلفن و سایر اطلاعات قابل شناسایی پنهان شده در داخل عکس می‌باشد. به تازگی ارتباطات بی‌سیم نیز دارای کد شناسایی در هدر بسته‌های شبکه‌ای شده‌اند که جزئیات منبعی که بسته‌ها از آن ارسال شده‌اند را دربر دارند.

رسوخ پنهان‌سازی داده‌ها در زندگی روزمره، از RFID‌های پنهان در محصولات که می‌خریم گرفته تا چاپگرهایی که در صفحه‌های چاپ‌شده‌ی آن‌ها اطلاعات قابل شناسایی ولی پنهانی وجود دارد، همه و همه بیانگر نفوذ داده‌های پنهان در زندگی روزمره تک‌تک ماست. حتی بسیاری از نرم‌افزارهایی که روزانه استفاده می‌شوند ویژگی‌هایی دارند که اجازه‌ی پنهان‌سازی داده‌ها را به کاربر می‌دهند. برای نمونه نرم‌افزار WinRar یکی از پرکاربردترین ابزارهای فشرده‌سازی در سیستم‌عامل‌های لینوکس و ویندوز مایکروسافت است و نکته‌ی جالب توجه این است که می‌تواند فایل‌های آرشیو آسیب‌دیده را ترمیم نماید. این ویژگی امکان پنهان کردن داده‌ها در فایل فشرده را فراهم می‌کند.



پنهان‌سازی داده‌ها، همسو با پیشرفت در سایر زمینه‌ها به روش‌های پیچیده‌تری به تکامل خود برای گریز از کشف و شناسایی ادامه می‌دهد. برخی از این روش‌ها، ترکیبی از دو یا چند مورد از تکنیک‌ها زیر می‌باشد: ۱- چندشکلی: دقیقاً همانند ویروس، برنامه‌های پنهان‌سازی داده‌ها نیز می‌توانند خوددگرگونی انجام داده تا هنگام شناسایی شدن بر اساس ردپا از چنگ پوششگرها بگریزند. ۲- چندبعدی: این نوع پنهان‌سازی داده از چند روش یا چند مرحله برای پنهان ساختن داده استفاده می‌کند. ۳- پراکندگی: برخی برنامه‌ها برای پنهان کردن داده‌ها از چند فایل حامل استفاده می‌کنند. از این گذشته، حتی ممکن است، فایل‌های تله‌ای برای سردرگم کردن سیستم‌های حفاظتی اضافه شود.

علیرضا اصغری

مرداد ۹۵

[Books2ara@yahoo.com](mailto:Books2ara@yahoo.com)

[Books2ara@gmail.com](mailto:Books2ara@gmail.com)

[Books2ara@outlook.com](mailto:Books2ara@outlook.com)



## پنهان سازی اطلاعات

### مقدمه

سه هزار سال است که بشر از روش‌های گوناگون پنهان سازی پیام‌ها برای فرستادن و دریافت اطلاعات، به عنوان بخش جدایی‌ناپذیری از جنگ‌ها استفاده می‌کند؛ به گونه‌ای که موفقیت یا شکست بسیاری از ماموریت‌های نظامی، به توانایی فرستادن و دریافت دستورها و ارتباط‌های سری و پوشیده بین واحدهای درگیر در جنگ بستگی دارد. امروزه، رویاروی انسان با مواردی چون ماموریت‌های جاسوسی در سطح بین‌الملل، برقراری ارتباط با مامور خودی مستقر در خاک دشمن، ارتباطات بین گروه‌های تروریستی و گروه‌های جنایتکار و تهدیدهای مداوم از ناحیه حملات سایبری، نیاز به برقراری ارتباطات از این دست را افزایش داده است. از سه هزار سال پیش تاکنون، اگرچه روش‌ها، فن‌ها و مفاهیم جدیدی در زمینه پنهان سازی داده‌ها ایجاد شده است، اما دگرگونی چندانی در اهداف برقراری چنین ارتباط‌هایی به وجود نیامده است.

در خلال دهه‌ی گذشته، پوشیده نگاری از پنهان سازی داده‌ها در عکس‌های دیجیتال، به سوی فایل‌های چندرسانه‌ای و پس از آن به سوی پروتکل‌های شبکه و در حال حاضر به سوی ابزارهای همراه هوشمند سوق یافته است. افزایش توانایی محاسباتی کامپیوترها و افزایش پهنای باند شبکه‌ها و همچنین ارتقای امکان برقراری ارتباط و حمل‌ونقل داده‌ها در ابزارهای ارتباطی همراه، موجب افزایش شتاب زندگی و در نتیجه نشت بیشتر اطلاعات شخصی در فضای مجازی و همچنین امکان برقراری ارتباط پنهان و سری در هر زمان و مکانی شده است.

این کتاب، با نگاهی گذرا به تاریخچه‌ی پوشیده نگاری، به بررسی تازه‌ترین تهدیدها، روش‌ها و فن‌های به‌کاررفته در پنهان سازی داده‌ها و ارتباطات سری می‌پردازد. همچنین با نگاهی به آینده و روش‌هایی که ممکن است در آینده ایجاد شود، به بررسی شیوه‌های کشف، تجزیه و تحلیل و آشکارسازی چنین روش‌هایی می‌پردازد.



# فصل اول

## تاریخچه رمزنگاری

### پیشگفتار:

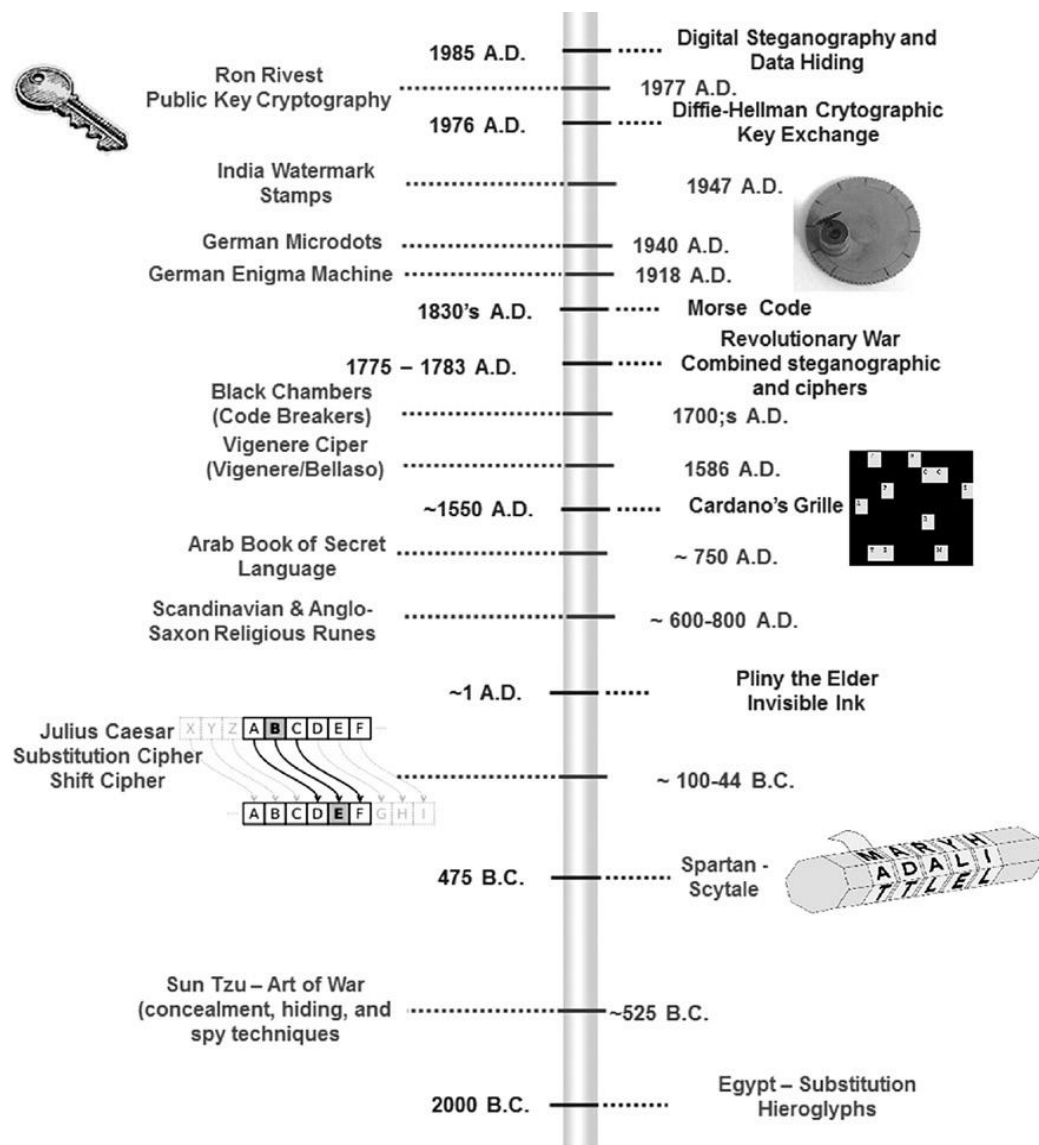
پنهان سازی داده‌ها، چه با اهداف خوب باشد و چه با اهداف بد، در تمام جنبه‌های زندگی ما خودنمایی می‌کند. همان‌گونه که بسیاری از مورخان، از جمله دیوید کوهن<sup>۱</sup> اذعان می‌کند، پنهان سازی داده‌ها هزاران سال پیش، از سری نویس منشعب شده و آغاز آن در تمدن مصر و به شکل حروف تصویری (هیروگلیف) بوده، که به عنوان نمادی از شروع عصر فراعنه تلقی می‌شود. در همان عصر، چینی‌ها از ابزارهای دیگری چون نوشتن پیام بر روی ابریشم یا کاغذ و به شکل کروی در آوردن آن و سپس آغشتنش به موم استفاده می‌کردند. پیام‌ها برای برقراری ارتباطات سیاسی یا انتقال دستورات یا اسرار نظامی به شکل پنهان به کار می‌رفتند. برای بالا بردن امنیت، پیام-رسان گوی را قورت می‌داد و به جای گوی مومی، خود را به مقصد می‌رساند. با پیشرفت تمدن و شهرنشینی اشکال ارتباط‌های پنهانی نیز پیچیده‌تر شده و برنامه‌های پیشرفته رمزنگاری و تغییر جایگاه حروف و کلمات در متن تولید شده و پیشرفت کردند.

مسلماً کتاب «کدشکنان»<sup>۲</sup> نوشته‌ی دیوید کوهن، مفصل‌ترین کتاب تاریخی در خصوص ارتباطات سری در اعصار گذشته است. شکل ۱-۱ برخی از اختراعات بشر در طول زمان، که بیشتر مربوط به مصر باستان و تمدن چین است را نشان می‌دهد.

---

<sup>۱</sup> David kohn

<sup>۲</sup> Codebreakers



شکل ۱-۱: تاریخچه‌ی پنهان سازی داده‌ها، استتار و در لفافه نوشتن.

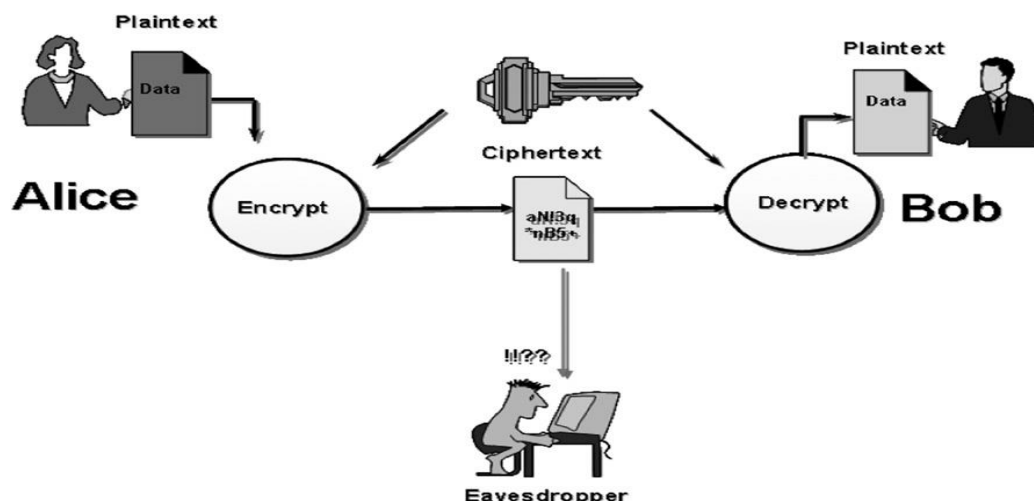
بنا به گواه تاریخ، سری نویسی از نیاز به ارتباطات پنهانی سرچشمه گرفته و ابزاری است که نظامیان برای حفاظت از ما در برابر تهاجم دشمن استفاده می‌کنند. دشمن نیز از همین ابزار برای تاخت‌وتاز به ما استفاده می‌کند. با پیشرفت فناوری، راه‌های بیشتری برای پنهان سازی داده‌ها در اختیارمان قرار می‌گیرد. امروزه، استفاده از این روش‌های پنهان سازی در جاسوسی صنعتی، تماس‌های جاسوسی، انتشار بد افزارها، استثمار کودکان و تروریسم رایج است. هر روزه، پوشیده نگاری‌های ویرانگر در پیرامون ما و در برابر چشمانمان اتفاق می‌افتد و بیشتر وقت‌ها هم کشف نمی‌شود.

امیدواریم بتوانیم در این کتاب اطلاعاتی درباره‌ی راه‌های گوناگونی که در پنهان سازی اطلاعات مورد استفاده قرار می‌گیرد، از ابزارهای فیزیکی گرفته تا ابزارهای دیجیتال، در اختیار شما قرار دهیم. اگرچه خطر فعالیت‌های جنایی هم وجود دارد، اما پنهان سازی داده‌ها همواره یک سرگرمی بسیار جالب بوده و حتی برای برخی از افراد، مثل شغل هست. اجازه دهید بحث را با مروری بر تاریخچه‌ی آنچه که ما را به پنهان سازی داده-های دیجیتالی رهنمون می‌شود، آغاز کنیم. برای این کار، نخست به فن‌های پیشینیان و پایه و اساس رمزنگاری و استتار نگاهی کوتاه می‌اندازیم.

## دانش رمزنگاری

برنامه‌های رمزنگاری و تغییر محل حروف و کلمات در کتاب‌های جورچین و روزنامه‌ها کاربرد زیادی دارند. رمزنگاری، یک حرف را جایگزین حرف دیگر، یا یک نماد را جایگزین نماد دیگری در متن می‌کند. هدف از رمزگشایی، مشخص کردن حرفی است که جایگزین حرف دیگر شده تا گیرنده پیام بتواند پیام اصلی را بازیابی کند. در روش رمزنگاری به شیوه‌ی تغییر محل واژه‌ها، حروف متن با حروف یا علائم دیگر جایگزین نمی‌شود و تنها محل قرار گرفتن حرف در همان کلمه جابجا می‌شود.

در هر یک از روش‌های بالا، پیام رمزی خروجی، روش یا الگوریتم ویژه‌ای است که باعث جایگزینی حروفی با حرف یا حروف و علائم دیگری یا تغییر محل قرار گرفتن حروف در متن پیام اصلی می‌شود و تنها فرستنده و گیرنده که از کلید آگاهی دارند، می‌توانند پیام را مشاهده نمایند و شخص دیگری نمی‌تواند پیام را خوانده یا از حالت رمز خارج نماید. معمولاً پیام رمز «Cipher» نامیده می‌شود. دشمن با استراق سمع نمی‌تواند پیام را بازیابی کند، مگر اینکه کلید و الگوریتم رمزنگاری را بداند. فرایند تحلیل رمز را «*cryptanalysis*» می‌نامند (شکل ۱-۲).



شکل ۱-۲: رمزنگاری

## رمزنگاری به شیوهی جایگزینی حروف

رمز جایگزین، روشی از رمزنگاری است که با استفاده از الگوریتم یا روش ویژه‌ای یک یا چند واژه در متن رمز، جایگزین یک واژه از متن اصلی می‌شود. از حروف، اعداد و نمادها و غیره می‌توان برای جایگزین متن اصلی استفاده کرد. الگوریتم رمزنگاری، تعیین‌کننده‌ی چگونگی جایگزینی بوده و بر پایه کلید است؛ بنابراین، گیرنده می‌بایست الگوریتم و کلید را برای رمزگشایی متن بداند. زمانی که گیرنده‌ی پیام رمزی را دریافت می‌کند، باید برای رمزگشایی پیام و آشکار کردن متن اصلی، الگوریتم جایگزینی واژه‌ها را هم بداند.

## رمز سزار

ژولیو سزار نخستین فردی بود که رمز جایگزینی را برای اهداف نظامی ابداع کرد. این روش رمزنگاری بدین ترتیب است که حروف یونانی را جایگزین حروف رومی می‌کرد و در نتیجه پیام برای دشمن ناخوانا و غیرقابل فهم می‌شد. بعدها روش سزار پایه ابداع روشی شد که امروزه به آن Shift Cipher می‌گویند. کاری که سزار برای رمزنگاری انجام می‌داد، جایگزینی حرف معینی به ازای هر حرف از حروف الفبا بود؛ سپس از این الفبای تغییریافته، در رمزنگاری متن استفاده می‌کرد. در این روش، نشانه‌های هر حرف الفبا با نمادهای متفاوتی جایگزینی می‌شود. به این شیوه از رمزنگاری، رمز تک الفبایی نیز می‌گویند. به عنوان مثال:

الفبای اولیه A B C D E F G H I J K L M N O P Q R S T U V W X Y Z



~~~~~  
 F G H I J K L M N O P Q R S T U V W X Y Z A B C D E<sup>۱</sup> الفبای رمزنگاری

با استفاده از این الفبای رمز، می‌توان متن را به شکل رمزی نوشت:

STEGANOGRAPHY RULES = پیام اصلی

XYJLFILWFUMD WZQJX = پیام رمزی

اگرچه با در نظر گرفتن توان محاسباتی موجود، این روش در استانداردهای رمزنگاری امروزی شیوهی ضعیفی محسوب می‌شود، اما هنوز هم روش اصلی سرگرمی‌های رمزگونه در روزنامه‌ها و حلقه رمزنگاری مخصوص کودکان (شکل ۱-۳) است.



شکل ۱-۳: حلقه رمزنگاری جانی کوست (Johnny Quest)

در جنگ جهانی دوم، ارتش آمریکا از اهالی قبیله Narajo در قالب روش ویژه‌ای از رمزنگاری استفاده کرد. در آن زمان، سرخ‌پوستان Narajo با گویش ویژه‌ای، که برای سایر افراد و دیگر قبایل سرخ‌پوست قابل فهم نبود، صحبت می‌کردند. در نتیجه ۲۹ قبیله Narajo، برای پشتیبانی از عملیات جنگی، در سپاه تفنگداران دریایی<sup>۲</sup> بکار گرفته شدند. سپاه تفنگداران دریایی از زبان Narajo به عنوان ابزار مطمئن ترجمه‌ی انگلیسی برای ارتباط در میدان جنگ استفاده می‌کرد. از آنجا که این گویش برای سایر افراد، جز اعضا قبیله Narajo و تعداد انگشت‌شماری از آمریکایی‌ها ناشناخته بود، در نتیجه جعل آن تقریباً غیرممکن بود.

<sup>۱</sup> هر حرف از الفبا جایگزین پنجمین حرف پس از خود شده است.

### کد پیام‌های رادیویی و کد مورس

در دهه ۱۸۳۰ ساموئل مورس کدی را برای ارسال پیام تلگرافی ابداع کرد. وی برای نشان دادن هر حرف، از تعداد معینی خط و نقطه استفاده می‌کرد. این کد که امروزه کد مورس نامیده می‌شود عبارت است از جایگزینی به روش سزار که هر حرف از الفبا و سایر علائم نگارشی با تعدادی مشخص خط و نقطه جایگزین می‌شدند (شکل ۱-۴).

|   |         |   |         |
|---|---------|---|---------|
| A | · —     | N | — ·     |
| B | — · · · | O | — — —   |
| C | — · — · | P | · — — · |
| D | — · ·   | Q | — — · — |
| E | ·       | R | · — ·   |
| F | · · — · | S | · · ·   |
| G | — — ·   | T | —       |
| H | · · · · | U | · · —   |
| I | · ·     | V | · · · — |
| J | · — — — | W | · — —   |
| K | — · —   | X | — · · — |
| L | · — · · | Y | — · — — |
| M | — —     | Z | — — · · |

شکل ۱-۴: جدول کد مورس

نمونه‌ای از شیوه‌ی رمز جایگزینی در کد مورس، در ترانه «Rush»، به نام «YYZ» استفاده شده است. جالب است که حروف YYZ کد فرودگاه تورنتو در کانادا، در نزدیکی شهر «Rush» است. در کد مورس، حرف «Y»، «— · —» و حرف «Z»، «— · — ·» است. با تبدیل «YYZ» به کد مورس، علائم به این شکل به دست می‌آید: «— · — · — · — ·». بسیاری از مردم نمی‌دانند که این علائم پایه و اساس مقدمه آهنگ هم هست.

برخی بر این باورند که کد مورس، رمزنگاری جایگزین نیست، زیرا هدف مورس پنهان ساختن پیام نبود، بلکه بیشتر استفاده از علائم خط و نقطه به عنوان شکلی از ارتباط در زمانی که هنوز تلفن اختراع نشده بود، می‌باشد. با این وجود، این روش، شکلی از رمز جایگزینی بوده و روشی از جایگزینی کد را نشان می‌دهد.

در خلال جنگ‌های اخیر، شکل‌های گوناگونی از این کد مورد استفاده قرار گرفته است. در واقع، اکثر مردم وقتی به آهنگ YYZ گوش می‌دادند، اصلاً تصور نمی‌کردند که این آهنگ با کد مورس آغاز می‌شود، بنابراین، می‌توان این روش را نوعی پنهان سازی اطلاعات تلقی کرد (پنهان کاری).

## رمزنگاری به روش Vigenore

در ابتدا، گروهی نخبه و روشنفکر روش رمزنگاری Vigenore را ابداع کردند، اما در نهایت فردی به نام Blaise de Vigenore آن را به صورت یک روش رمزنگاری ساماندهی نمود. در این روش، به جای این که جایگزینی بر اساس تنها یک حرف از حروف الفبا باشد، جایگزینی بر تمام ۲۶ حرف الفبا مبتنی بود (شکل ۱-۵).

|   | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| B | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A |
| C | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B |
| D | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C |
| E | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D |
| F | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E |
| G | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F |
| H | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G |
| I | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H |
| J | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I |
| K | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J |
| L | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K |
| M | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L |
| N | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M |
| O | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N |
| P | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| Q | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
| R | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q |
| S | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R |
| T | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
| U | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
| V | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U |
| W | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V |
| X | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
| Y | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X |
| Z | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y |

شکل ۱-۵: جدول رمزنگاری به روش vigenore

استفاده از تنها یک ستون در جدول رمزنگاری Vigenere مشابه رمزنگاری به روش سزار است. بنابراین، جدول Vigenere طوری طراحی شده است که در آن از چندین ردیف استفاده شود و برای هر حرفی که باید رمزگذاری شود، ردیفی جداگانه مورد استفاده قرار می‌گیرد. این کار به وسیله‌ی استفاده از کلمه‌ی کلید

اختصاص در فرآیند رمزنگاری صورت می‌گیرد. به عنوان مثال، اگر کلمه Combo را به عنوان کلید انتخاب و از جدول Vigenere استفاده کنیم، می‌توانیم عبارت زیر را به این ترتیب رمزنگاری کرد.

پیام: thekeyisunderthedoormat

گذر واژه: combo

متن رمز: vvqlsawevbfsduvgamu

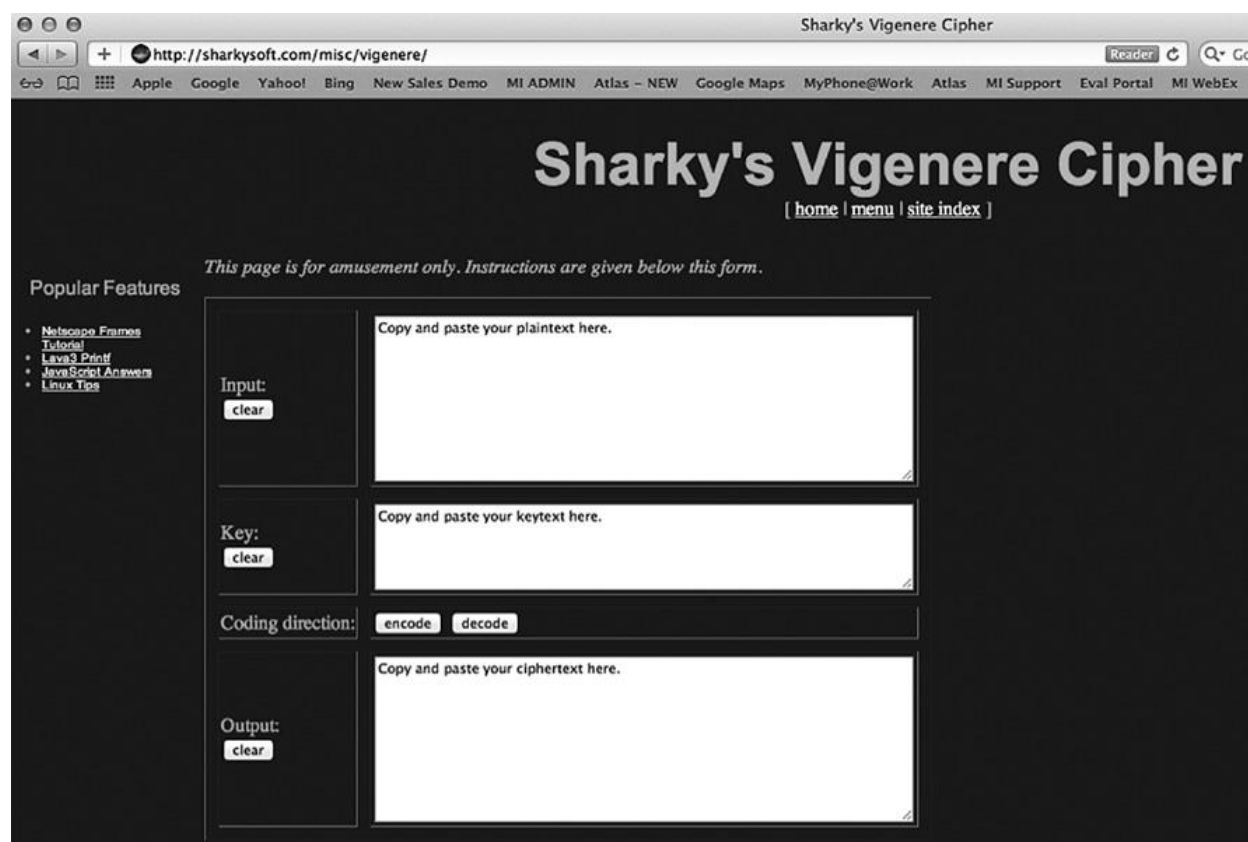
این شیوه از رمز جایگزینی را چند الفبایی می‌نامند، زیرا برخلاف رمزنگاری تک الفبایی سزار، از چند حرف الفبای متفاوت برای هر حرف از الفبا در رمزنگاری استفاده می‌شود. رمز Vigenere در زمان انتشارش غیرقابل شکستن بود. به عنوان مثال، در کد سزار، رمز شکن می‌توانست با بررسی تعداد تکرار حروف، یعنی با توجه به این که حروف e و n بیش‌ترین و حروف X و Z کمترین احتمال تکرار در واژه‌ها را دارند، پیام را رمزگشایی کند. شکل ۱-۶ فرکانس تکرار حروف زبان انگلیسی را به ترتیب از بالاترین تکرارپذیری تا پایین‌ترین آن نشان می‌دهد.

|           |        |         |        |
|-----------|--------|---------|--------|
| High      | Medium | Low     | Rare   |
| ETAONIRSH | DLUCM  | PFYWGBV | JKQXZ  |
| Highest   |        | →       | Lowest |

شکل ۱-۶: تکرار حروف زبان انگلیسی

علاوه بر تحلیل تعداد تکرار حروف، رمز شکن از ویژگی‌های زبانشناسی برای رمزگشایی متن نیز استفاده می‌کرد. به عنوان مثال، ترکیب «io» در کلمات انگلیسی بسیار معمول است، در صورتی که ترکیب «oi» بسیار کمیاب است. رمز شکنان در گذشته از لیستی از کلمات که هیچ‌وقت باهم در کلمات بکار نمی‌رفتند، برای حذف ترکیبات کمیاب بهره می‌جستند. برای بهره‌مندی از این ویژگی فرض بر این است که رمز شکن زبانی را که پیام به آن نوشته شده است را می‌داند، درحالی‌که همیشه چنین نیست. اگر رمز شکن نداند زبان بکار رفته در پیام فرانسه است یا اسپانیایی یا زبانی دیگر، در نتیجه نمی‌تواند از این ویژگی استفاده نماید. بنابراین تشخیص زبان بکار رفته در متن رمزنگاری، برای رمز شکن بسیار ضروری و حیاتی است.

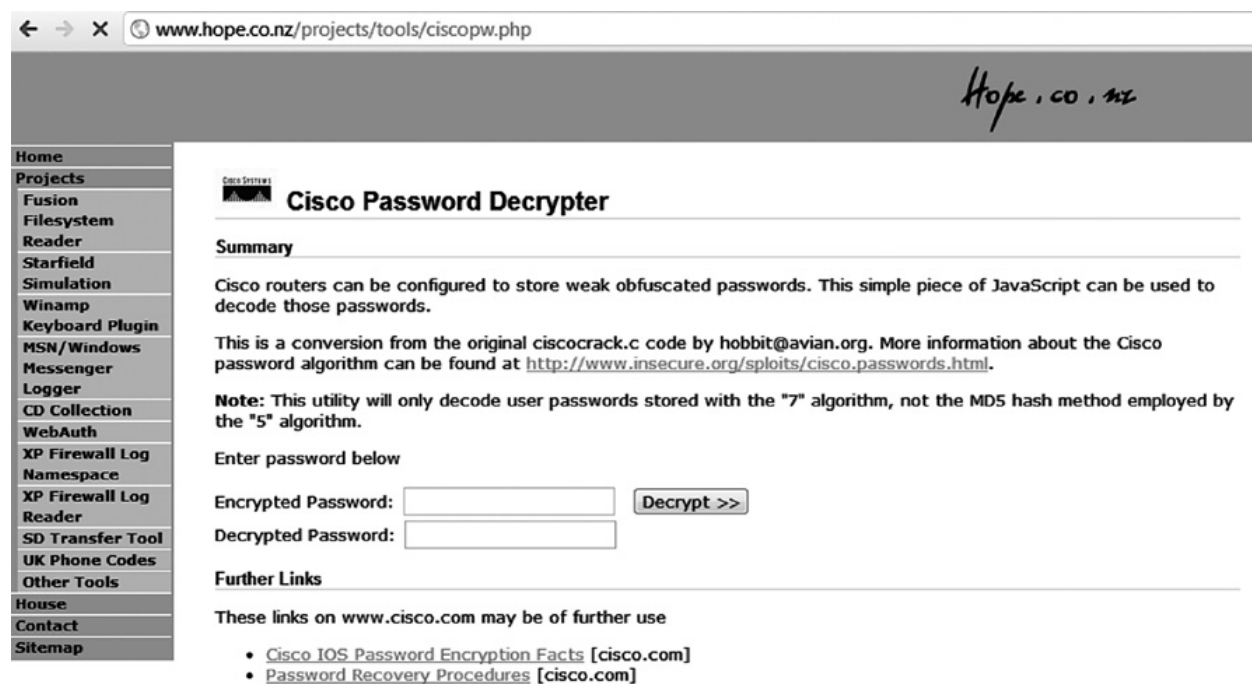
از آنجا که در رمزنگاری به شیوه Vigenere از کلیدهای بیشتری برای جایگزینی حروف استفاده می-شود، بنابراین امکان استفاده از ویژگی تکرار حروف در کلمه یا علائم زبان‌شناسی برای رمزگشایی پیام، در عمل وجود ندارد. در ضمن، رمزشکن در رمزنگاری به روش Vigenere پیچیدگی دیگری نیز دارد: تخمین تعداد کلیدهای ممکن و طول کلید. به همین دلیل بود که رمزنگاری Vigenere برای صد سال یعنی تا سال ۱۸۴۵ غیرقابل شکستن باقی ماند. در این سال Charles Babbye توانست تحلیل موفقی را برای رمزگشایی شیوه رمزنگاری Vigenere انجام دهد. امروزه، ابزارهای گوناگونی در وب برای نوشتن پیام‌های رمزی با استفاده از رمزنگاری Vigenere وجود دارد. این ابزارها را می‌توان به سادگی در اینترنت یافت، بنابراین تقریباً همه می‌توانند پیامی را رمزنگاری کنند (شکل ۷-۱).



شکل ۷-۱: ابزار رمزنگاری به شیوه Vigenere در اینترنت

## الگوریتم دایجیست

اگرچه با در نظر گرفتن استانداردها و قدرت محاسباتی کامپیوتر امروزی، رمزنگاری Vigenere روش ضعیفی است، اما همچنان در بسیاری از پیاده‌سازی‌ها به کار می‌رود. به عنوان مثال، نسخه ترکیبی از رمزنگاری Vigenere را می‌توان در سیستم‌عامل شرکت سیسکو<sup>۱</sup> و در روترها و ابزارهای شبکه یافت. گرچه استفاده از درهم‌ساز MD5<sup>۲</sup>، یک گزینه‌ی پشتیبانی در IOS است، اما هنوز بسیاری از دستگاه‌های سیسکو از کلمه عبور ۷ حرفی هشینگ (ترکیبی از رمزنگاری Vigenere) استفاده می‌کنند. ابزارهای بسیاری برای رمزگشایی کلمه عبور ۷ نویسه‌ای در سیستم‌عامل سیسکو وجود دارد. بنابراین، به ناظران شبکه‌های کامپیوتری اکیداً توصیه می‌شود که سازوکار درهم‌ساز پیش‌فرض را از کلمه‌ی عبور ۷ حرفی به MD5 تغییر دهند، چرا که نقاط ضعف رمزنگاری Vigenere کاملاً شناخته شده است (شکل ۸-۱).



شکل ۸-۱: نرم‌افزار رمزشکن گذر واژه‌ی روتر شرکت سیسکو در اینترنت

<sup>۱</sup> Cisco IOS

<sup>۲</sup> Message Digest Algorithm

## رمزنگاری به روش تغییر محل حروف

شکل دیگری از شیوه‌های رمزنگاری، رمزنگاری با استفاده از تغییر محل حروف است. روش کار آن، تغییر محل قرار گرفتن حروف در متن اصلی است. این شیوه رمزنگاری در جورچین‌ها و روزنامه‌ها به وفور استفاده شده و به نام‌های Jumble , anayram هم معروف است به عنوان مثال:

متن رمز      Hiddenmessage => dihegassemned      متن اولیه

این گونه از رمزنگاری به روش تغییر محل حروف را می‌توان به سادگی رمزگشایی کرد. اجازه دهید به نمونه‌ی پیچیده‌تری از رمزنگاری بپردازیم.

## Spartan Scytale

شاید یکی از کهن‌ترین روش‌های شناخته شده رمزنگاری به روش تغییر محل حروف در واژه‌ها، « Spartan Scytale » (که معمولاً به آن Scytale می‌گویند) باشد. در یونان باستان (حدود ۴۷۵ پیش از میلاد)، فرماندهان ارتش Scytale را برای ارسال پیام‌های سری اختراع کردند (شکل ۱-۹). برای رمزنگاری فرماندهان ارتش پوست بز یا چرم را به دور Scytale چوبی می‌پیچیدند؛ سپس پیام سری را در طول آن می‌نوشتند؛ سپس آن را باز کرده و برای فرمانده گیرنده پیام می‌فرستادند. در طول مسیر اگر پیام به دست دشمنی می‌افتاد، بدون در اختیار داشتن Scytale چوبی مناسب، پیام بی‌معنی بود و بیشتر توالی‌ای از حروف به نظر می‌رسید. فرمانده گیرنده، چرم را به دور Scytale مناسب می‌پیچید و پیام اصلی آشکار می‌شد. این روش تغییر محل، یکی از بدوی‌ترین روش‌های رمزنگاری به روش تغییر محل است.



شکل ۱-۹: Spartan Scytale

اگر محل قرار گرفتن پیامی را که بیشتر حروفش جابه‌جا شده‌اند را عوض کنیم، یا به عبارت دیگر تغییر محل دوگانه حروف، روشی برای بالا بردن سطح پیچیدگی برای پیشگیری از رمزگشایی پیام است.

### تفاوت رمزنگاری به روش تغییر محل حروف و رمزنگاری به روش تغییر شکل حروف

تغییر محل حروف با تغییر شکل حروف متفاوت است. در تغییر محل حروف، حروف متن اصلی بدون تغییر باقی می‌ماند، فقط محل آن‌ها یا به عبارتی، ترتیب آن‌ها عوض می‌شود؛ درحالی‌که در روش تغییر حروف، ترتیب حروف تغییر نمی‌کند، بلکه شکل حروف تغییر می‌کنند. همان‌گونه که پیشتر اشاره شد، رمزنگاری به روش تغییر محل حروف محدود به اصل تغییر مکان می‌باشد؛ به همین دلیل به سادگی می‌توان حتی به صورت دستی و بدون استفاده از قدرت محاسباتی کامپیوتر، راه‌های گوناگونی برای تغییر مکان حروف در متن پیدا کرد. این تغییر مکان‌ها، به هزاران روش قابل پیاده‌سازی و اجراست و برخی از آن‌ها پیچیدگی‌های بالایی نیز دارند.

امروزه، با پیدایش کامپیوتر، پیچیدگی رمز جایگزینی به طرز چشمگیری افزایش یافته است. همچنین، افزایش قدرت پردازش کامپیوترها امکان ترکیب روش تغییر محل و تغییر شکل حروف در یک روش رمزنگاری را به صورت همزمان فراهم می‌کند. به عنوان مثال، استاندارد رمزنگاری داده‌ها<sup>۱</sup>، ۱۶ مرحله تغییر محل و تغییر شکل هر گروه هفت‌تایی از حروف را همزمان اجرا می‌کند، چیزی که صدها سال پیش غیرممکن بود. امروزه کامپیوتر، قدرتمندترین سلاح در دست رمزشکنان نیز می‌باشد.

### پوشیده‌نگاری

معمولاً مردم بین همپوشانی در تعریف رمزنگاری و استتار گیج می‌شوند. برداشت مردم از استتار، نوشتن به صورت مخفیانه یا سری است. این برداشت از نظر فنی نادرست است و ناشی از معنای کلمه یونانی «Crypst» در مقابل «Steagn» است؛ به عبارت دیگر، تفاوت بین مفهوم «نوشتن پنهان» و «نوشتن در پوشش» می‌باشد.

در رمزنگاری، نوشتن پنهان، به روشی از نگارش داده‌ها گفته می‌شود که برای چشم غیرمسلح قابل مشاهده بوده ولی بدون تجزیه و تحلیل بی‌معنی باشد. /ستتار، نوشتن به صورتی است که توسط چشم غیرمسلح قابل مشاهده نباشد و به آن نوشتن نامرئی یا در لفافه نیز گفته می‌شود.

<sup>1</sup> Data Encryption Standard (DES)



این سردرگمی می‌تواند ناشی از تعریف کلمه انگلیسی «hidden» نیز باشد. در فرهنگ واژگان Random House، این کلمه به معنی «پنهان، مبهم، نهان» آمده است؛ در نتیجه، قابل درک است که چرا مردم هنگام توصیف «رمزنگاری» و «استتار» دچار نوعی سوءتفاهم و کج فهمی می‌شوند. این تعریف می‌تواند حاکی از نوعی همپوشانی بین کاربرد «رمزنگاری» و «استتار» باشد، در صورتی که در اصل چنین نیست. برای یافتن تفاوت بین «رمزنگاری» و «استتار»، از خود بپرسید که آیا پیام به هم ریخته است یا پنهان؟ اگر پیام به هم ریخته بود، رمزنگاری شده و اگر پنهان بود، استتار شده است.

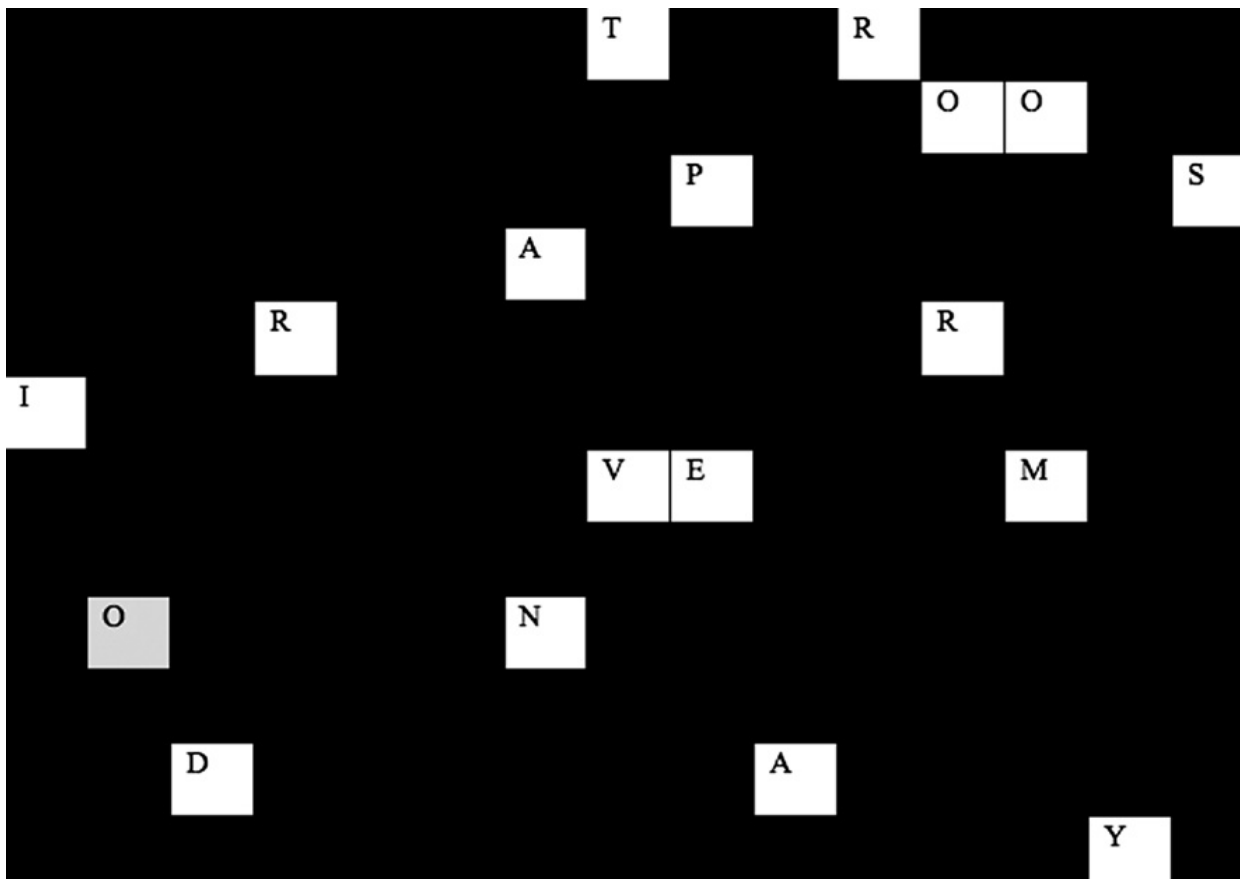
### Cardano's Grille

Cirdam Gardan ایتالیایی، اولین رمز شبکه‌ی مشبک را ابداع کرد. این شیوه‌ی رمزنگاری شامل استفاده از ورق محکمی از جنس کاغذ، فلز یا هر چیز دیگر است که با برش بخش‌هایی به ظاهر تصادفی، اما برنامه ریزی شده و هدفمند روی این ورق است. به این صفحه، شبکه مشبک می‌گویند. یک پیام به ظاهر عادی می‌تواند دارای توالی حروف و کلماتی باشد که به صورتی هدفمند در مکان‌های ویژه‌ای قرار گرفته‌اند، در نتیجه در همین پیام عادی، می‌تواند پیام دیگری مستتر باشد. به عنوان مثال، شکل ۱-۱۰ در نگاه نخست، متنی عادی به نظر می‌رسد.

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | H | E |   | W | E | A | T | H | E | R |   | H | A | S |
| B | E | E | N |   | V | E | R | Y |   | C | O | O | L |   |
| L | A | T | E | L | Y | . |   | P | E | R | H | A | P | S |
|   | T | H | E |   | F | A | L | L |   | W | I | L | L |   |
| S | T | A | R | T |   | S | O | O | N | E | R |   | T | H |
| I | S |   | Y | E | A | R | . |   | P | R | A | C | T | I |
| C | A | L | L | Y |   | E | V | E | R | Y |   | M | A | P |
| L | E |   | H | A | S |   | S | T | A | R | T | E | D |   |
| T | O |   | C | H | A | N | G | E |   | C | O | L | O | R |
| . |   | W | E |   | H | A | V | E |   | F | I | N | I | S |
| H | E | D |   | O | U | R |   | L | A | S | T |   | H | A |
| R | V | E | S | T |   | O | F |   | T | H | E |   | Y | E |
| A | R | . |   |   |   |   |   |   |   |   |   |   |   |   |

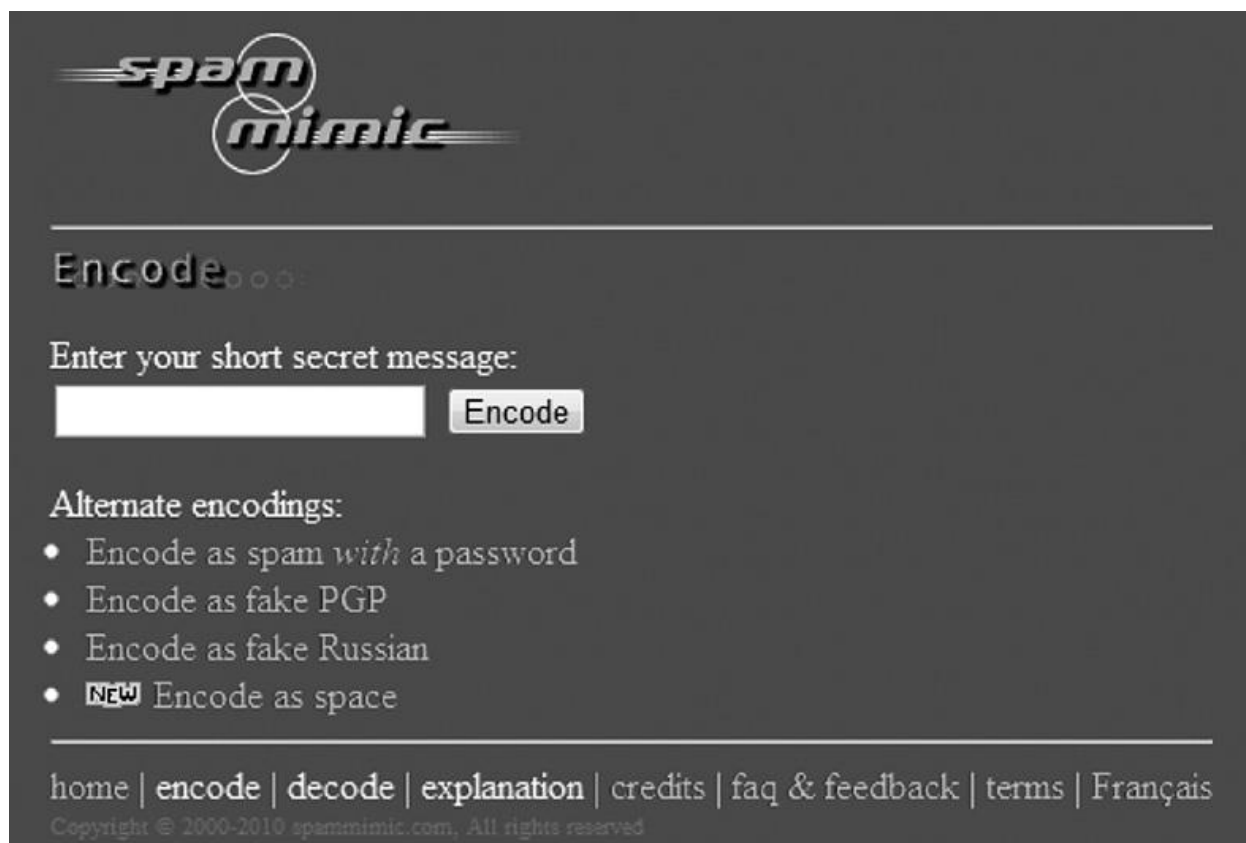
شکل ۱-۱۰: Cardano's Grille با پیام پنهان

اما اگر گیرنده بداند که چه مکانی در متن اصلی را برای یافتن متن پنهانی جستجو کند، می‌تواند با استفاده از Cardano's Grille، پیام پنهان شده در دل پیام اصلی را پیدا کند. بدین صورت که با قرار دادن ورقه‌ی مشبک روی متن اصلی، پیام پنهان را بخواند. در این حالت، همان‌گونه که در شکل ۱-۱۱ نشان داده شده است، پیام پنهانی به این صورت آشکار می‌گردد: «لشگر دوشنبه می‌رسد».



شکل ۱-۱۱: Cardano Grille با ورق مشبک جهت آشکار ساختن پیام مشبک

رمز Cardano Grille امروزه هم کاربرد دارد، به عنوان مثال وب‌گاه‌های مولد «هرزنامه‌ها» از این روش استفاده می‌کنند. اما به جای استفاده از ورق مشبک برای آشکارسازی پیام پنهان، یک برنامه کامپیوتری، کار رمزنگاری و رمزگشایی را انجام می‌دهد (شکل ۱-۱۲).



شکل ۱-۱۲: رمزنگاری در هرزنامه‌ها

در اینجا هدف، ایجاد پیامی است که یک هرزنامه به نظر برسد، اما در واقع در دل آن، پیامی رمزنگاری شده نهفته است. مردم روزانه تعداد بیشماری هرزنامه دریافت می‌کنند. اگر شخصی از محل پیام پنهان آگاه باشد، می‌تواند پیام را آشکار کند، درحالی‌که سایر افراد بدون این که بدانند که هرزنامه پیام پنهانی در دل خود دارد، به سادگی به عنوان هرزنامه از کنار آن می‌گذرند.

### جوهر نامرئی

نخستین نمونه‌ای استفاده از جوهر نامرئی به قرن اول میلادی بر می‌گردد، زمانی که Pliny درباره کشف خود در خصوص شیرهی نوعی کاکتوس به نام «tithymalus» و امکان استفاده از آن برای نوشتن نامرئی مطالبی را عنوان نمود. این نوشته، نخستین سند در زمینه پنهان سازی پیام (استتار) تلقی می‌شود.

شاید شناخته‌شده‌ترین نوع جوهر نامرئی، آب لیمو باشد، که از آن برای نوشتن متن بر روی کاغذ استفاده می‌شود و پس از خشک شدن برای چشم غیرمسلح نامرئی می‌شود، اما در مجاورت منبع گرما، مانند گرمای حباب لامپ، متن به آرامی آشکار می‌گردد. جوهرهای دیگری که ترکیب اسیدی دارند، مانند سرکه، شراب، آب پیاز، شیر، ادرار و آب بارانی که با اسید سولفوریک ترکیب شده باشد، هنگامی که در معرض گرما قرار می‌گیرند اکسیده شده و متن نوشته شده را در معرض دید قرار می‌دهند.

کتاب ساموئل رابین (۱۹۸۷) تحت عنوان «علم اسرار آمیز جوهرهای پنهان»<sup>۱</sup> شاید جامع‌ترین کتاب در این زمینه باشد. این کتاب به فن‌های جوهر نامرئی محرمانه CIA، که توسط سازمان غیرانتفاعی معروف به «پروژه جیمز مدیسون»<sup>۲</sup> فاش شده است، می‌پردازد. برای دریافت فهرستی از این دستورالعمل‌ها به وب سایت <http://www.jamesmadisonproject.org> مراجعه کنید. شایان یادآوری است که برخی از سازمان‌های دولتی خاص بر این باورند که برخی از این فرمول‌ها باید سری و دور از دسترس عموم باشد. به همین دلیل برخی از این فرمول‌ها در این کتاب منتشر نشده است.

## ریزخال‌ها<sup>۳</sup>

به سختی می‌توان موضوع استتار را بدون اشاره به ریزخال‌ها تکمیل کرد. این خال‌ها، در واقع عکس‌های کوچک شده‌ای به اندازه‌ی یک نقطه بر روی یک صفحه چاپی هستند. این خال‌ها می‌تواند یک نقطه مثلاً روی حرف «i»، یا به هر شکل دیگری روی صفحه باشد.

اگرچه ایده‌ی ریزفیلم‌ها به سال ۱۸۷۰ در پاریس برمی‌گردد، اما در سال ۱۹۴۰ میلادی F.B.I به وسیله مامور دوجانبه‌اش متوجه شد که آلمان‌ها این روش را برای تولید ریزخال‌ها به کمال رسانده‌اند. در سال ۱۹۴۱ بود که بالاخره F.B.I توانست اولین ریزخال را روی پاکت یک مامور آلمانی تحت تعقیب کشف کند. پس از آن نوارهای باریکی از فیلم در زیر تمبر پاکت کشف شد، که بعدها مشخص شد برای اهداف جاسوسی از آن استفاده شده است. اطلاعات دزدیده شده فراوانی، از جمله داده‌های طراحی اورانیوم، آمار تولید، برنامه‌های ساخت،

<sup>۱</sup> The Secret Science of Covert Inks

<sup>۲</sup> James Madison Project

<sup>۳</sup> Microdots

دیگرام‌های آن و غیره کشف شد. در آن زمان، جاسوسان از دوربین کوچکی بنام «Minox» برای عکسبرداری از اسناد استفاده می‌کردند (شکل ۱-۱۳).



شکل ۱-۱۳: دوربین Minox Spy

مراحل تولید ریز خال‌ها شامل گرفتن چاپ بسیار کوچکی از عکس دوربین و عکس گرفتن دوباره از آن به وسیله‌ی میکروسکوپ معکوس می‌باشد. با این کار می‌توان ابعاد عکس تا قطر ۰/۰۵ اینچ کاهش می‌یابد. بعدها این روش توسعه یافت و از سوزن تزریق زیرجلدی برای منتقل کردن و چکاندن ریز خال‌ها روی قسمت مشخصی از نامه‌ی تایپ‌شده استفاده کردند. پس از انتقال ریزخال بر صفحه، آن را به وسیله‌ی ماده‌ی شیمیایی Coklion در محل خود تثبیت می‌کردند. گیرنده، ابزارهای گوناگونی برای مشاهده‌ی محتویات ریزخال در اختیار داشت.

تنها مشکل استفاده از ریز خال‌ها این بود که جوهری که ریز خال‌ها را با آن می‌نوشتند بسیار براق بود؛ در نتیجه، نامه‌ای که گمان می‌رفت ریزخال در آن مورد استفاده قرار گرفته است را مستقیماً جلو نور می‌گرفتند؛ در زاویه خاصی، جوهر ریزخال شروع به درخشیدن می‌کرد، درحالی‌که جوهر معمولی این‌گونه نبود.

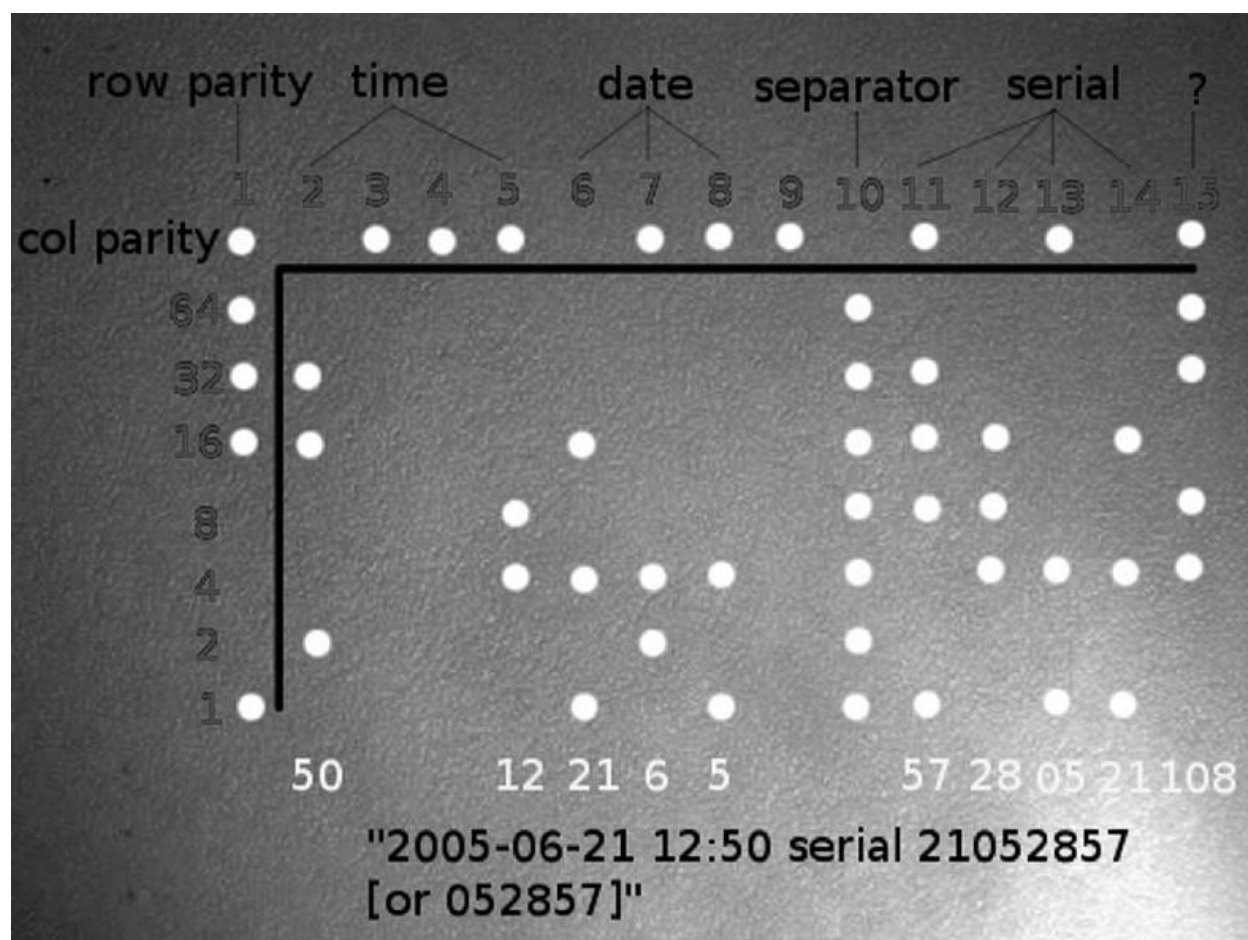
اگرچه مخترع واقعی ریزخال معلوم نیست، اما شخصی بنام پروفیسور Zapp، که مخترع دوربین بسیار کوچک Minox است، را مبدع ریزخال می‌دانند. در جنگ جهانی دوم سازمان جاسوسی انگلستان کیت‌های ریزخال‌ها را به نام تجهیزات Zapp می‌نامیدند.

شایان توجه است که فناوری ریزخال برای ارسال کل سند و مدرک مناسب است، درحالی‌که جوهر نامرئی بیشتر برای ارسال پیام‌های کوتاه به کار می‌رود. از ریزخال‌ها می‌توان برای ارسال اسناد و مدارکی که حاوی اشکال و نمودارها است، استفاده نمود، در صورتی‌که جوهر نامرئی، این قابلیت را ندارد. امروزه ریزخال‌ها هم در تولید تراشه و هم در کارخانه‌های اتومبیل‌سازی برای تایید اصالت خودرو کاربرد دارد.

## کاربرد ریزخال‌ها در چاپگرها

در سال ۲۰۰۴ مجله PCworld مقاله‌ای منتشر کرد و در آن هشدار داد که سازندگان چاپگر در هر صفحه‌ای که چاپ می‌کنند یک ریزخال زرد را نیز چاپ می‌کنند.

بنیاد تازه‌های الکترونیک<sup>۱</sup> این گزارش را پیگیری و رمز به‌کاررفته در چاپگر Daco Color شرکت زیراکس را رمزگشایی کرد. این بنیاد کشف کرد، که خال زرد، شماره‌ی سریال چاپگر، تاریخ و ساعت چاپ را مشخص می‌کند. نکته جالب این است که ریزخال‌ها توسط چشم غیرمسلح قابل مشاهده نیست، اما زیر نور آبی و با استفاده از ذره‌بین می‌توان ریزخال پنهان را مشاهده کرد (شکل ۱-۱۴).



شکل ۱-۱۴: ریزخال‌های ره‌گیری چاپ

<sup>1</sup> Electronic Frontier Foundation

همان‌گونه که در شکل ۱-۱۵ نشان داده شده است، بنیاد تازه‌های الکترونیک صفحه مشبک آشکارسازی رمزهای ریزخال‌ها چاپ را رمزگشایی کرد.

|            | 1                     | 2                     | 3                     | 4                     | 5                     | 6                     | 7                     | 8                     | 9                     | 10                    | 11                    | 12                    | 13                    | 14                    | 15                    |
|------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| col parity | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 64         | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 32         | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 16         | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 8          | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 4          | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 2          | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |
| 1          | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> |

شکل ۱-۱۵: برنامه رمزگشایی ریزخال چاپگر DocuColor به وسیله‌ی بنیاد تازه‌های الکترونیک

باور بیشتر مردم درباره‌ی ریزخال‌ها این است که برای مقاصد قانونی ساخته شده و در مراجع قضایی کاربرد دارد. بنیاد تازه‌های الکترونیک لیستی از پرینترهای شناخته‌شده‌ای که این خال‌ها را چاپ می‌گیرند، جمع‌آوری کرده و در نشانی <http://www.eff.org/pages/list-printers> می‌توانید لیست سازندگانی که از ریزخال‌ها استفاده می‌کنند را مشاهده نمایید. اگر چاپگر شما در این لیست نبود، می‌توانید از امکان رمزگشای آنلاین برای رمزگشایی ریزخال‌های آن استفاده کنید.

## واترمارک

واترمارک، روشی است مشابه تکنیک «استتار». قرن‌هاست که از این فن در چاپ اسکناس و تمبر برای تشخیص اصل و تقلبی بودن آن‌ها استفاده می‌شود. هدف اصلی واترمارک خلق تصویری شفاف بر روی کاغذ برای تأیید اصالت آن است. از آنجایی که قرن‌ها پیش هزینه‌ی ارسال نامه به وسیله‌ی پست بالا بود، استفاده از تمبرهای تقلبی بین مردم امری عادی به شمار می‌آمد. به عنوان مثال، در هندوستان برای جلوگیری از تقلب و جعل تمبر، با استفاده از واترمارک تصویر شفافی از یک فیل روی تمبرها حک می‌شد.

در زمان چاپ اسکناس، واترمارک‌های گوناگونی به آن اضافه می‌شود. به عنوان مثال، اکثر اسکناس‌ها در ایالات متحده، با واترمارکی از یک عکس، چاپ شده‌اند. به عنوان نمونه، اگر اسکناس ۱۰۰ دلاری را جلوی نور بگیرید، تصویر بنجامین فرانکلین را می‌توانید مشاهده نمایند (شکل ۱-۱۶).



شکل ۱-۱۶: عکس واترمارک در اسکناس ۱۰۰ دلاری آمریکا

برای حفظ مالکیت معنوی و اثبات اصالت رسانه‌های دیجیتالی مثل محصولات دیداری، شنیداری و موسیقی از واترمارک دیجیتال استفاده می‌شود.

شایان یادآوری است که اگرچه واترمارک و استتار از نظر جاسازی داده‌ها شباهت‌های زیادی باهم دارند، اما هدف اصلی واترمارک مشکل ساختن دستیابی به داده‌های جاسازی شده نیست، بلکه هدف اصلی آن مشکل کردن حذف واترمارک و در نتیجه جلوگیری از استفاده غیرقانونی از فایل می‌باشد.



### چکیده:

ارتباطات سری، تاریخچه‌ای جالب در تمدن‌ها، جنگ‌ها و فرهنگ‌های گوناگون دارد. امروزه بسیاری از روش‌های برجسته رمزنگاری و ارتباطات سری دوران گذشته، به شیوه‌ای ظریف به دنیای دیجیتالی امروز نیز راه یافته است. در این کتاب سفر استتار داده‌های دیجیتال، به ما بپیوندد تا بسیاری از جدیدترین روش‌های استتار داده-ها در تمام سیستم‌های عامل، دستگاه‌های موبایل، فایل‌های چندرسانه‌ای و سایر قالب‌های دیجیتال را بررسی - کنیم.

~~~~~

## فصل دوم

### چهار روش ساده‌ی پنهان‌سازی داده‌ها

بسیاری از نرم‌افزارهایی که روزانه استفاده می‌شوند ویژگی‌هایی دارند که اجازه‌ی پنهان‌سازی داده‌ها را به کاربر می‌دهند. برای نمونه در نرم‌افزار ورد شرکت مایکروسافت، کاربر می‌تواند از گزینه Property برای تعیین نام مؤلف، نام شرکت، کلمات کلیدی متن و بسیاری داده‌های دیگر استفاده نماید و به این‌گونه داده‌ها، ابر داده گفته می‌شود. در صورت ارسال فایل به کاربر دیگر، او نیز می‌تواند این ویژگی‌ها را به همان شیوه و برای اعمال تغییرات به کار گیرد؛ در نتیجه نرم‌افزار ورد می‌تواند کاربر ایجاد کننده فایل، زمان ایجاد فایل و تغییرات در فایل را کنترل کند و این کار را به وسیله‌ی ابرداده‌ی جداگانه‌ای انجام می‌دهد که به صورت خودکار به فایل اصلی ضمیمه می‌شود. این پروسه به لحاظ حفظ مسائل امنیتی هنگامی که فایل بارها به خارج از سازمان انتقال می‌یابد و یا به وب‌سایتی پست می‌شود در نظر گرفته شده تا سازمان بتواند رد فایل-هایی را که سهواً به خارج از سازمان منتقل شده‌اند را بگیرد. چنانچه مشخصات شما، شرکت، شماره تلفن، نشانی پست الکترونیک و یا احتمالاً اطلاعات حساسیت برانگیزی از شما در معرض دید دیگران بر روی اینترنت قرار گیرد، چه حالی و وضعی پیدا می‌کنید؟

دولت آمریکا نگران این شیوه از انتشار اسناد بوده و روش‌ها و مراحل ویژه‌ای را برای پاک‌سازی فایل-ها، پیش از انتشار و دسترسی همگانی به اسناد، پیش‌بینی کرده است. برای مثال سازمان امنیت ملی آمریکا (NSA) مقاله‌ای تحت عنوان « ابرداده و داده‌های پنهان در فایل‌های PDF، ریسک انتشار و اقدام متقابل » منتشر کرد و روش پاک‌سازی فایل‌های PDF پیش از قرار دادن آن‌ها بر روی وب یا ارسال به سازمان‌های دیگر را مشخص نموده است. این روش پاک‌سازی، نه تنها ابرداده‌ها را شامل می‌شود، بلکه لایه‌های پنهان در مهندسی متن، متون مبهم و عکس‌ها را نیز شامل می‌شود.

با استفاده از ابزاری به نام FOCA<sup>1</sup> می‌توانید صفحات وب و خدمات آنلاین که حاوی داده‌ها و ابرداده‌های جالب هستند را پوشش کنید. به این وسیله می‌توانید حجم گسترده‌ای از اطلاعات که به وسیله فایل‌های ورد، PDF و صفحات اینترنتی به بیرون نشت می‌کند را شاهد باشید. ابزارهای انتشاراتی بسیاری به کاربران اجازه می‌دهند تا نام مؤلف، نام شرکت، جزئیات اطلاعات شرکت، عناوین، متن، Tag و غیره را ثبت کنند و ابزارهایی مانند FOCA می‌تواند تمامی این داده‌ها را گردآوری کرده و شرکت مبدأ و یا موسسه‌ی دولتی متبوع را شناسایی کند. بسیاری از سازمان‌ها از وجود این‌گونه داده‌ها بی‌خبر بوده و ندانسته امکان انتشار آن را در اینترنت فراهم می‌کنند.

## پنهان‌سازی داده‌ها در نرم‌افزار ورد

نرم‌افزار ورد شرکت مایکروسافت، الگوی برتر واژه‌پردازهای دیگر است، در واقع بسیاری از کاربران کامپیوتر MAC، از واژه‌پرداز ورد استفاده می‌کنند و این موضوع می‌تواند دلیل کافی برای بررسی راه‌هایی که می‌توان اطلاعات را در فایل‌های ورد پنهان نموده باشد.

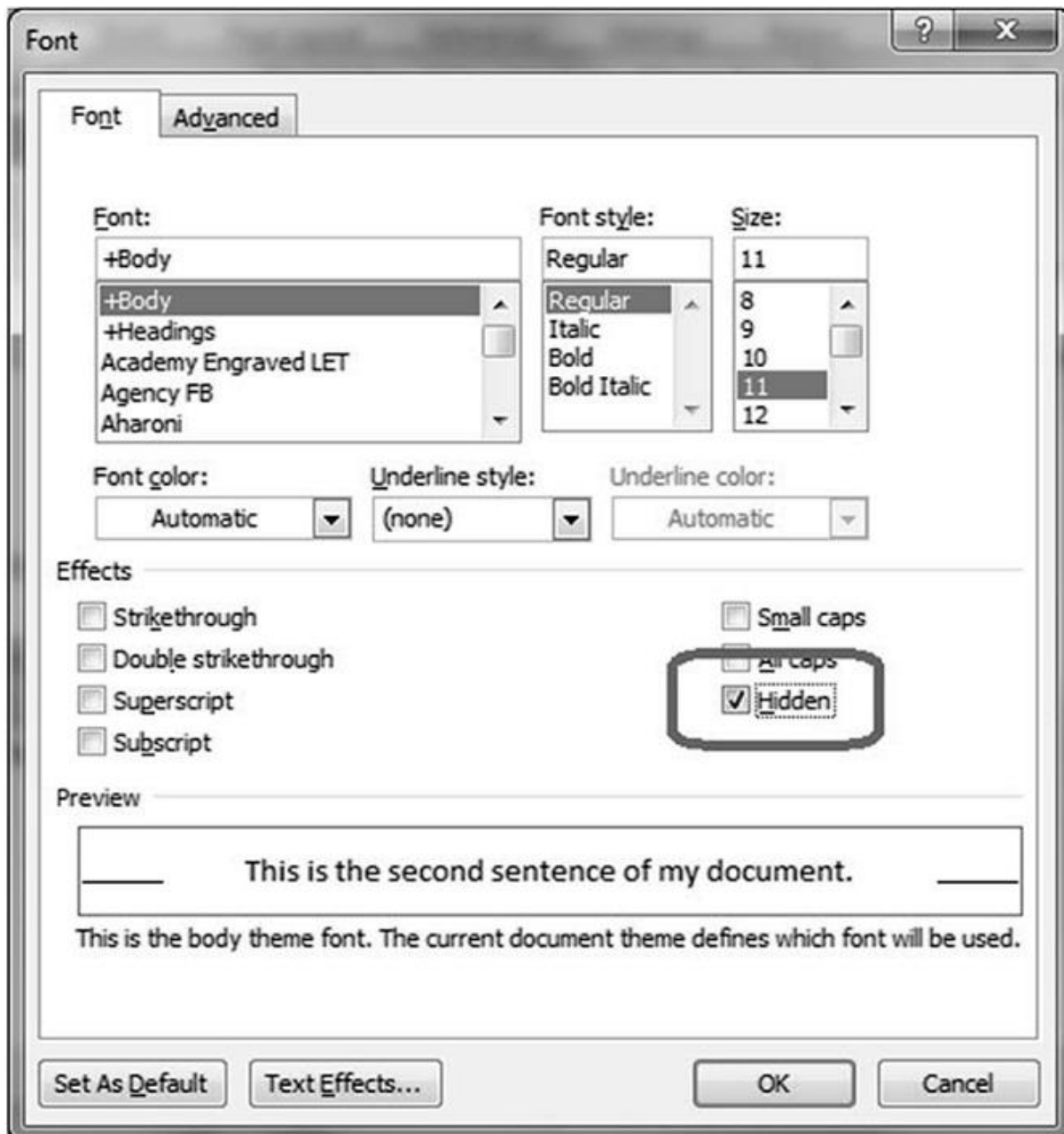
نرم‌افزارهای ورد و اکسل شرکت مایکروسافت راه‌های گوناگونی برای نهان‌سازی داده‌ها در اختیار ما قرار می‌دهند که شامل فیلد توضیحات، اطلاعات شخصی، واترمارک، محتویات نامرئی و داده‌های XML ویژه‌ی کاربران می‌شود. استفاده از گزینه Hidden که یکی از امکانات گزینه Font است، راه ساده و سرگرم‌کننده‌ای برای پنهان کردن متن است. برای پنهان کردن متن در فایل ورد نخست متن را به شکل معمول تایپ نمایید، سپس متنی را که می‌خواهید پنهان شود را به متن اضافه کنید (عکس ۲-۱).



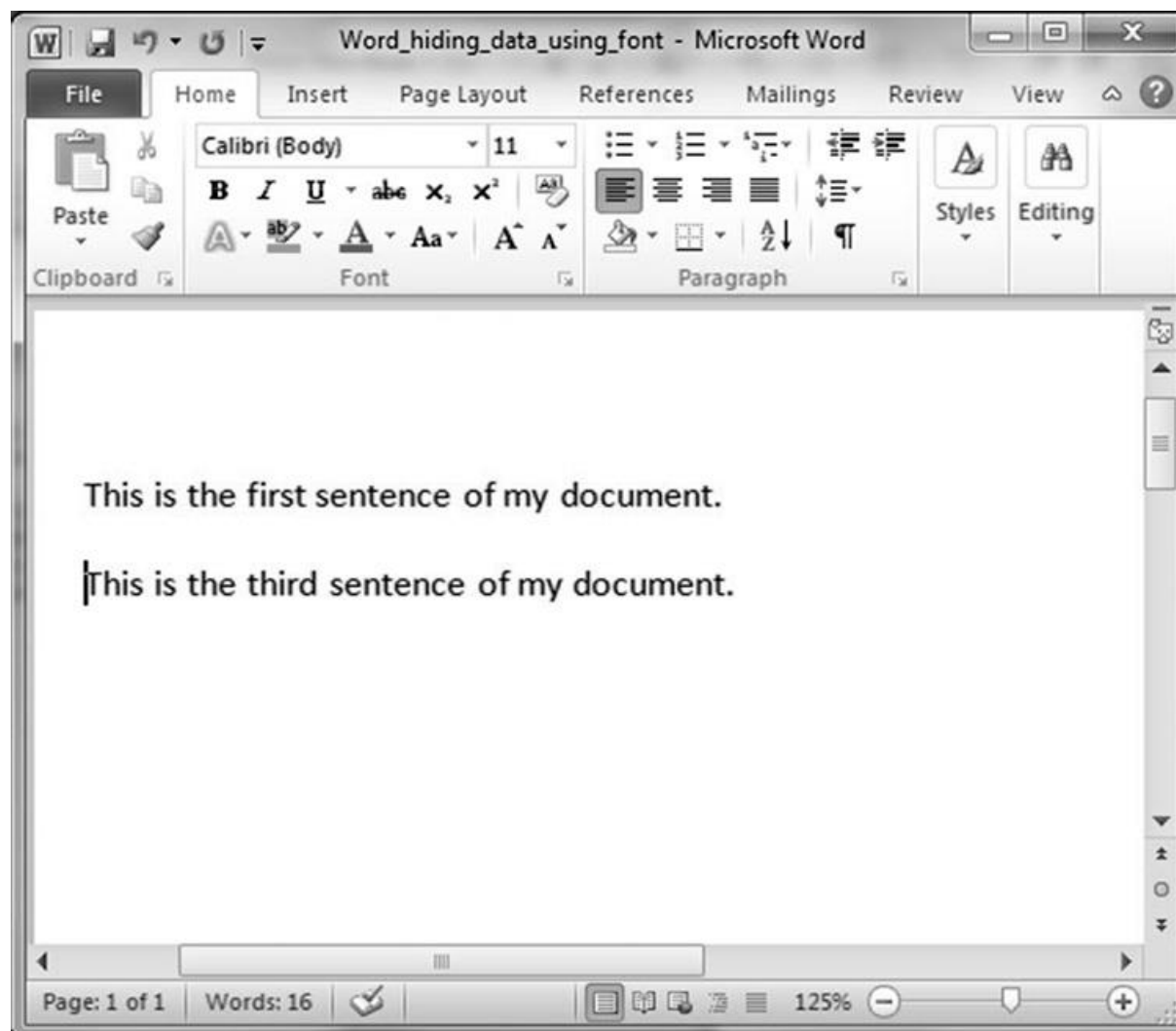
شکل ۲-۱: چگونگی پنهان‌سازی متن در نرم‌افزار مایکروسافت ورد

<sup>1</sup> Fingerprinting Organizations and Collected Archives

متن مورد نظر خود که می‌خواهید پنهان نمایید را انتخاب نموده و از منوی font گزینه Hidden را برگزینید، سپس فایل را ذخیره نمایید. با این کار متن پنهان شده در مشاهده‌ی فایل به شکل عادی دیده نخواهد شد (عکس ۲-۲ و ۳-۲).

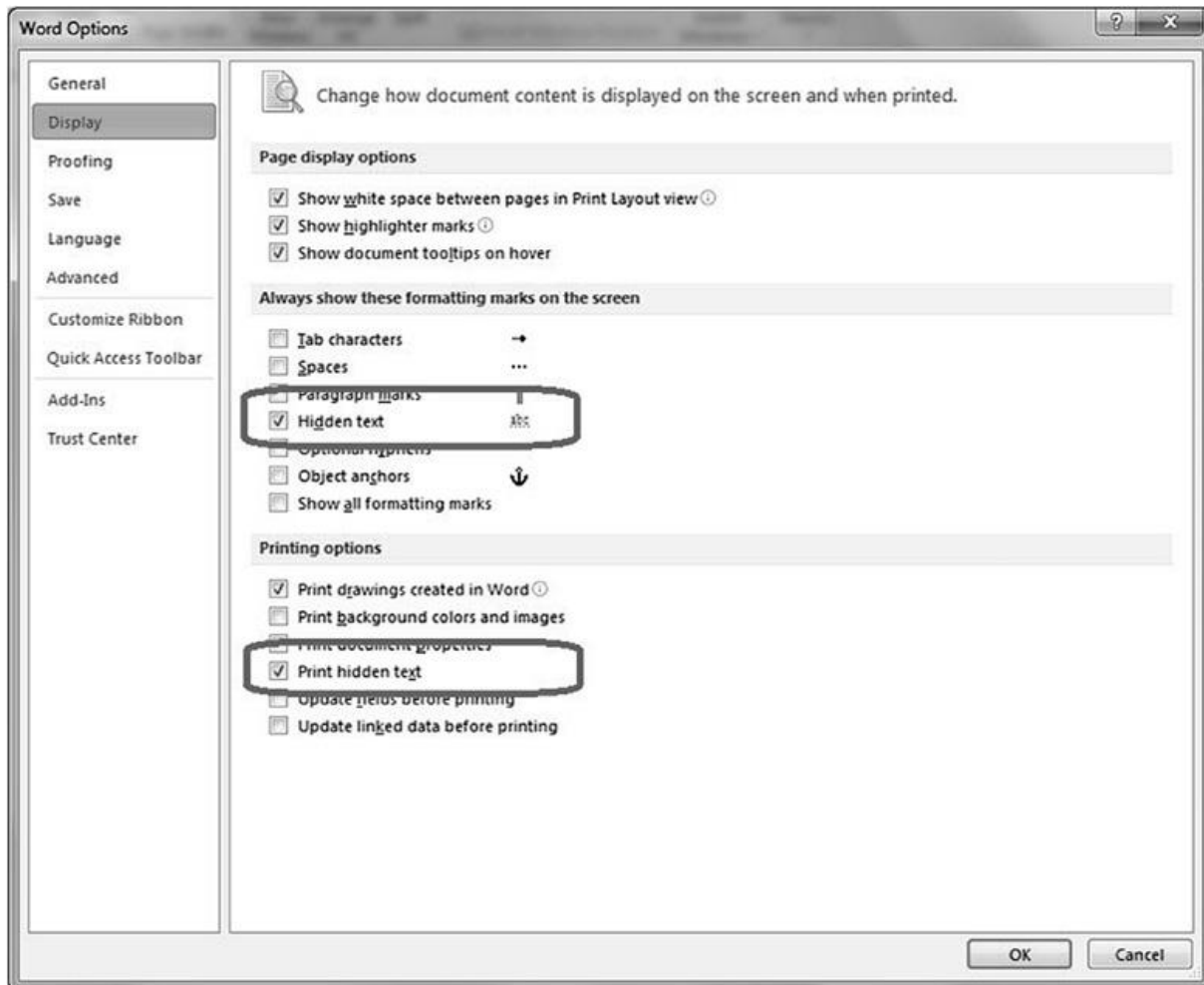


شکل ۲-۲: استفاده از امکان پنهان‌سازی در ورد مایکروسافت



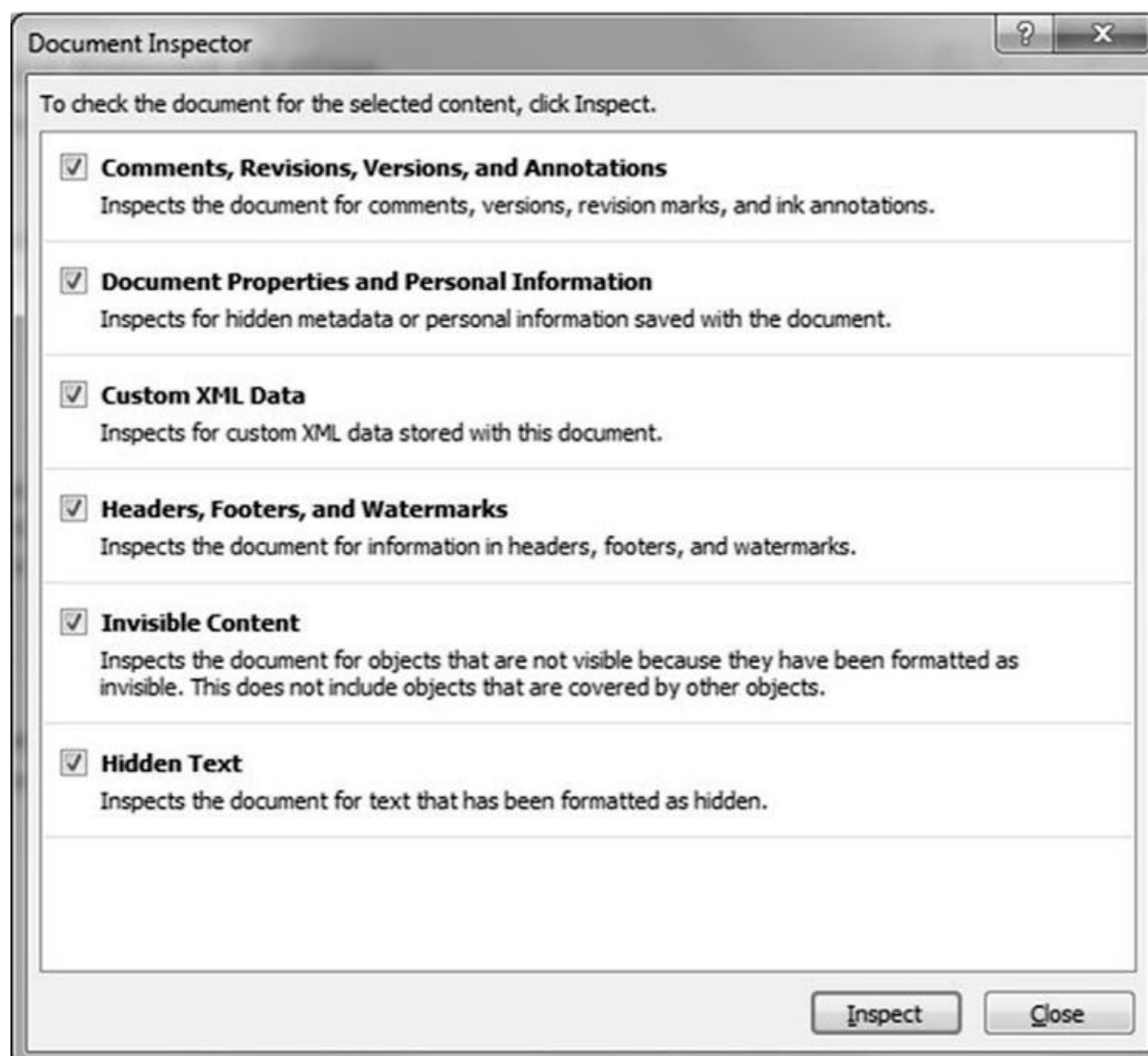
شکل ۲-۳: پنهان‌سازی جمله دوم در ورد مایکروسافت

به طور پیش‌فرض، متن مخفی در هنگام چاپ نیز چاپ نمی‌شود. برای مشاهده‌ی متن پنهان از منوی File -> option -> Display و سپس گزینه‌ی Hidden را فعال کنید. با این کار امکان ویرایش متن پنهان را پیدا می‌کنید و برای چاپ آن نیز گزینه‌ی Print را فعال نمایید (عکس ۲-۴).



شکل ۲-۴: امکانات ورد مایکروسافت برای نمایش متون پنهان شده در سند

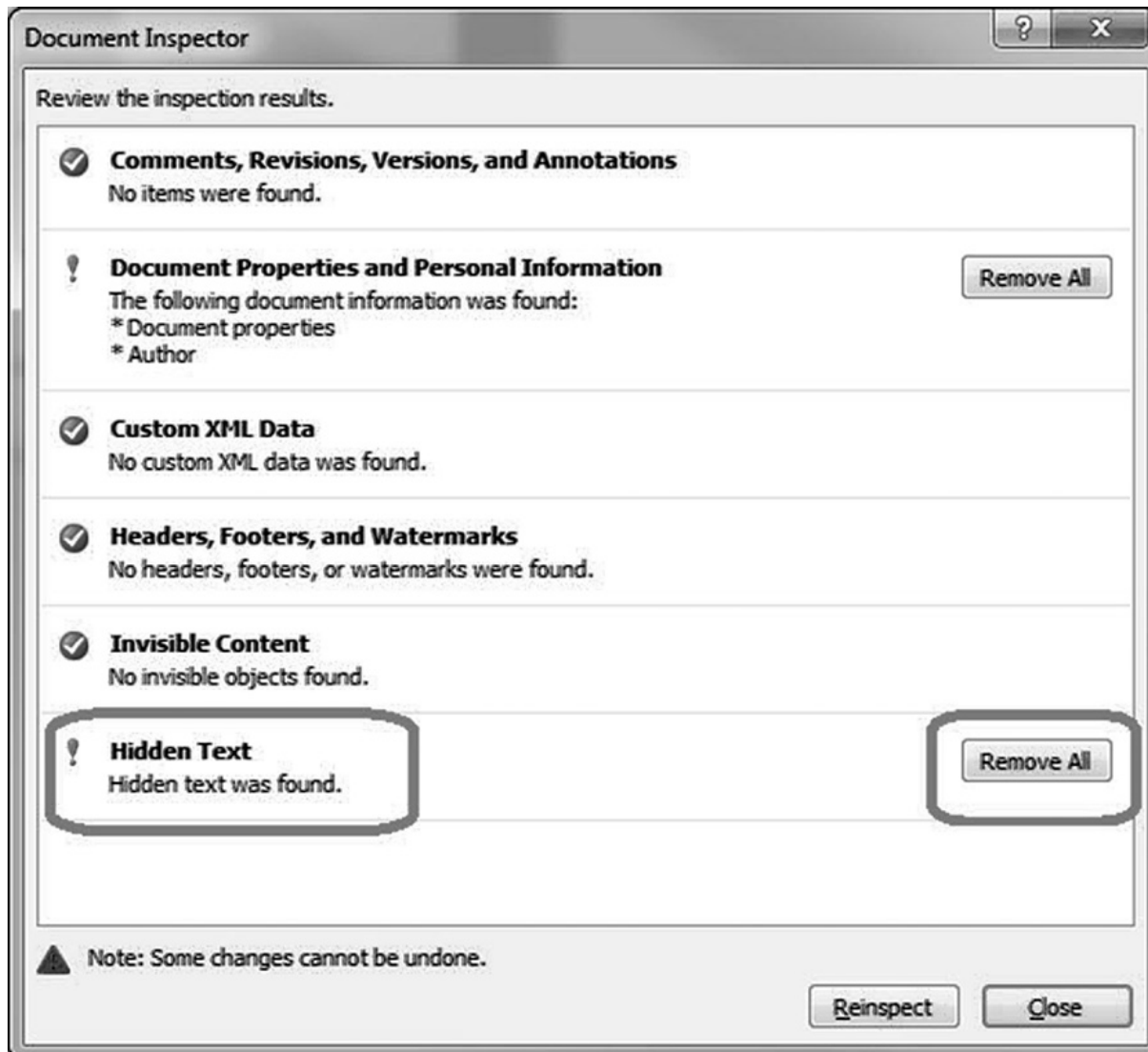
راه دیگر کشف وجود متن پنهان در فایل ورود استفاده از گزینه‌ی **File => Info => Check for Issues => Inspect Document** می‌باشد. استفاده از گزینه‌ی **inspect** یک روش عالی برای مشاهده انواع ابر داده‌های پنهان در متن، از قبیل مشخصات مؤلف، توضیحات و سایر اطلاعات هویتی شخص نگارنده متن و در کنار آن‌ها، امکان آشکار کردن متون پنهان در سند می‌باشد (عکس ۲-۵).



شکل ۲-۵: استفاده از Document Inspector برای یافتن ابرداده‌ها و داده‌های پنهان در متن

استفاده از Inspector منتهی به آشکار شدن ابرداده‌ها و تولید گزارش نتیجه بازرسی می‌شود و کاربر در صورت تمایل می‌تواند متن پنهان شده را حذف نماید. نکته جالب در این خصوص این است که مردم غالباً وجود متن پنهان را بررسی نمی‌کنند و از وجود آن در متن هم ناآگاهند (عکس ۲-۶).





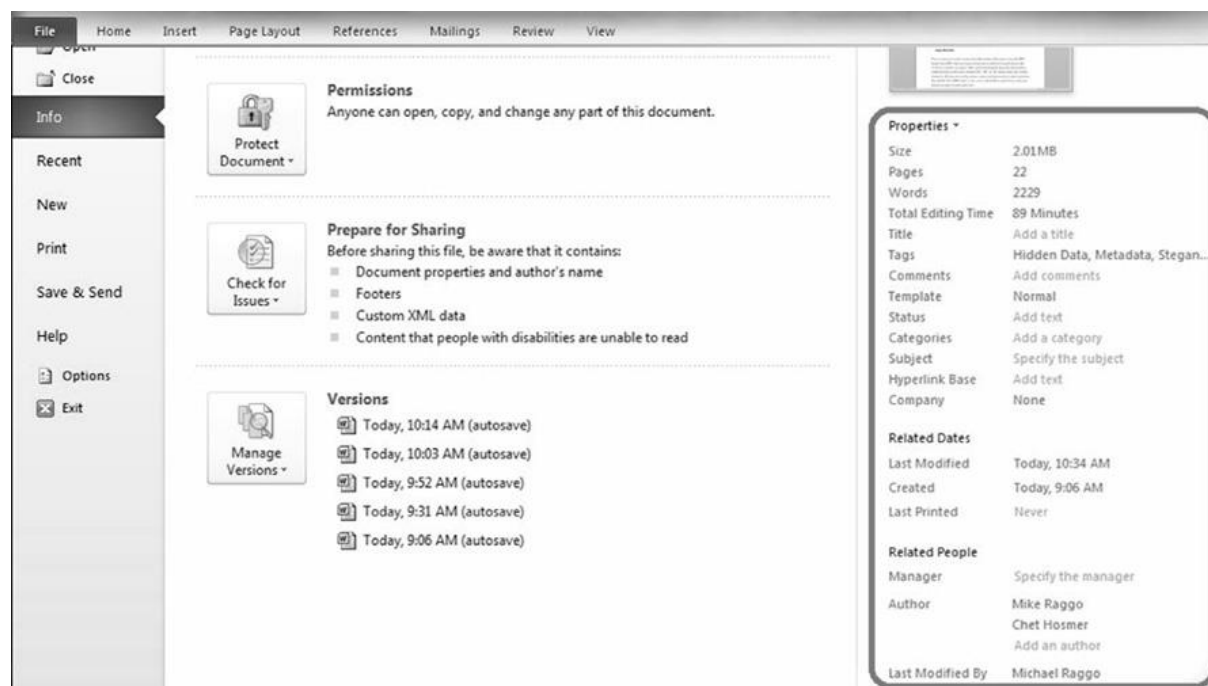
شکل ۲-۶: امکان کشف و حذف داده‌ی پنهان در نرم‌افزار ورد

نکته مهم در این روش پنهان‌سازی این است که تنها راه دستیابی به متن پنهان شده، استفاده از گزینه Font است. حال اگر کاربر، اقدام به تایپ متن با رنگ سفید بر روی کاغذ سفید نماید، متن قابل مشاهده نبوده و توسط inspect هم قابل تشخیص نیست.

یکی از کاربردهای متن پنهان زمانی است که شما بخواهید دو نسخه از یک متن را چاپ کنید، یکی شامل متن پنهان و دیگری بدون آن، همین کاربرد در مورد ارائه پاورپوینت متصور است. نمایش ارائه بدون متن پنهان برای کاربران و دیگر ارائه با توضیحات برای سخنران.

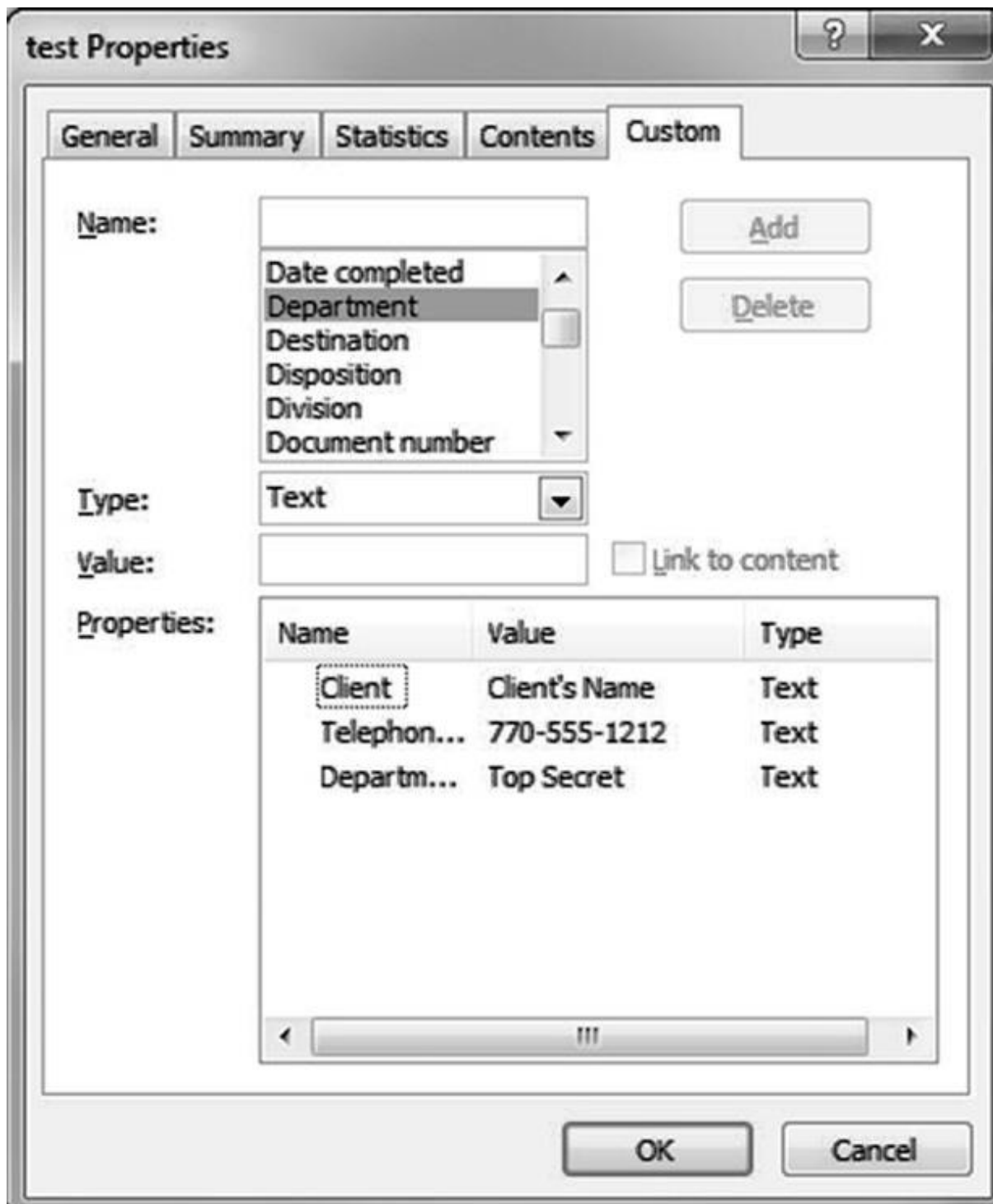
موارد گوناگون دیگر را می‌توان در گزینه Advanced properties همانند شکل ۲-۷ مشاهده

نمایید.



شکل ۲-۷: ابر داده‌ها و properties ورد مایکروسافت

اگر بخواهید آیتم مورد نظر خود را به Properties اضافه کنید، از گزینه Advance و انتخاب از لیست Dropdown همانند شکل ۲-۸ آیتم مورد نظر خود را اضافه کنید.



شکل ۲-۸: امکانات گزینه Advanced در اضافه کردن آیتم‌های مورد نظر کاربر در ورد مایکروسافت

به این نکته توجه کنید که آیتم‌های افزوده شده در مشاهده‌ی استاندارد گزینه‌های Properties نمایش داده نمی‌شوند و فقط به صورت دستی و با انتخاب گزینه Custom در منوی Advance قابل مشاهده است.

## ابرداده‌های در فایل تصاویر

سازمان‌هایی مثل FBI و وزارت دادگستری، از عکس‌ها برای ره‌گیری و دستگیری جنایتکاران استفاده می‌کنند. هنگامی که جنایتکاری اقدام به انتقال عکس با استفاده از اینترنت می‌نماید و یا عکس را به وب‌سایتی پست می‌کند، این سازمان‌ها با استفاده از ابرداده‌های همراه عکس، اقدام به شناسایی محل عکسبرداری می‌نمایند. قاتلی به نام مستعار BTK و با نام واقعی دنیس رادر<sup>۱</sup> در طول سی سال اقدام به کشتن چند نفر کرد و پلیس محلی را با سرنخ‌های ساختگی در قتل‌ها گمراه کرده بود. کلید ارتباط مدارک مربوطه قاتل سریالی، از فلاپی دیسک ارسالی از سوی او سرچشمه گرفت. بازرسان قضایی اقدام به بازیابی فایل‌های حذف شده از دیسکت کرده و یک فایل ورد را به همراه ابرداده آن بازیابی کردند. ابرداده نشان داد که فایل به وسیله، شخصی ساکن کلیسا به نام دنیس ایجاد شده است. تصادفاً کلیسایی وجود داشت که شخصی به نام دنیس به عنوان سرپرست کلیسا در آن کار می‌کرد. تحقیقات بعدی و مدارک دیگری از قبیل DNA ثابت کرد که این شخص قاتل ۱۰ نفر بوده است.

ابزارهای بسیاری برای دستکاری ابرداده‌های عکس‌ها به ویژه فایل‌ها در قالب JPEG وجود دارد. از آنجایی که قالب JPEG همگانی‌ترین قالب مورد استفاده در دوربین‌ها و موبایل‌هاست در نتیجه تحلیل خود را بر روی این قالب متمرکز می‌کنیم. EXIF شامل استاندارد بسیاری از انواع قالب‌های عکس مثل JPEG و TIFF و غیره است و چگونگی عملکرد سرایند و Tag عکس‌ها را برای تعیین پارامترهای فایل در زمان عکسبرداری به وسیله دوربین، اسکنر و ابزارهای دیگری که حاوی ابرداده‌هایی در دل فایل اصلی هستند را مشخص می‌کند و این سرایند امکان پنهان کردن داده‌ها برای کاربران ویژه، برای گریز از تشخیص بسیاری از ابزارهای امنیت شبکه فراهم می‌سازد. گوگل نرم‌افزار رایگانی به نام Picasa ارائه کرده که امکان مشاهده و تغییر سرایند استاندارد EXIF را دارد. وقتی که فایلی را در این نرم‌افزار باز کنید Properties آن امکان مشاهده داده‌هایی چون تاریخ، نوع دوربین (دوربین عکاسی یا موبایل) تاریخ و زمانی که عکس گرفته شده و سایر اطلاعات شناسایی عکس را فراهم می‌کند. بسیاری از مردم پیش از به اشتراک گذاری یا ارسال عکس آن به وسیله اینترنت، اقدام به پاک کردن این‌گونه داده‌ها می‌کنند، زیرا برخی دوربین‌ها امکان ثبت موقعیت GPS را نیز داشته و با نمایان شدن این اطلاعات، محل دقیق مکان عکسبرداری قابل شناسایی است؛ در نتیجه بسیاری به لحاظ حفظ حریم خصوصی، این داده‌ها را پاک می‌کنند (عکس ۲-۹).

<sup>1</sup> Dennis Roder



عکس ۲-۹: مشاهده و تغییر داده های properties و هدر EXIF در نرم افزار Picasa شرکت گوگل

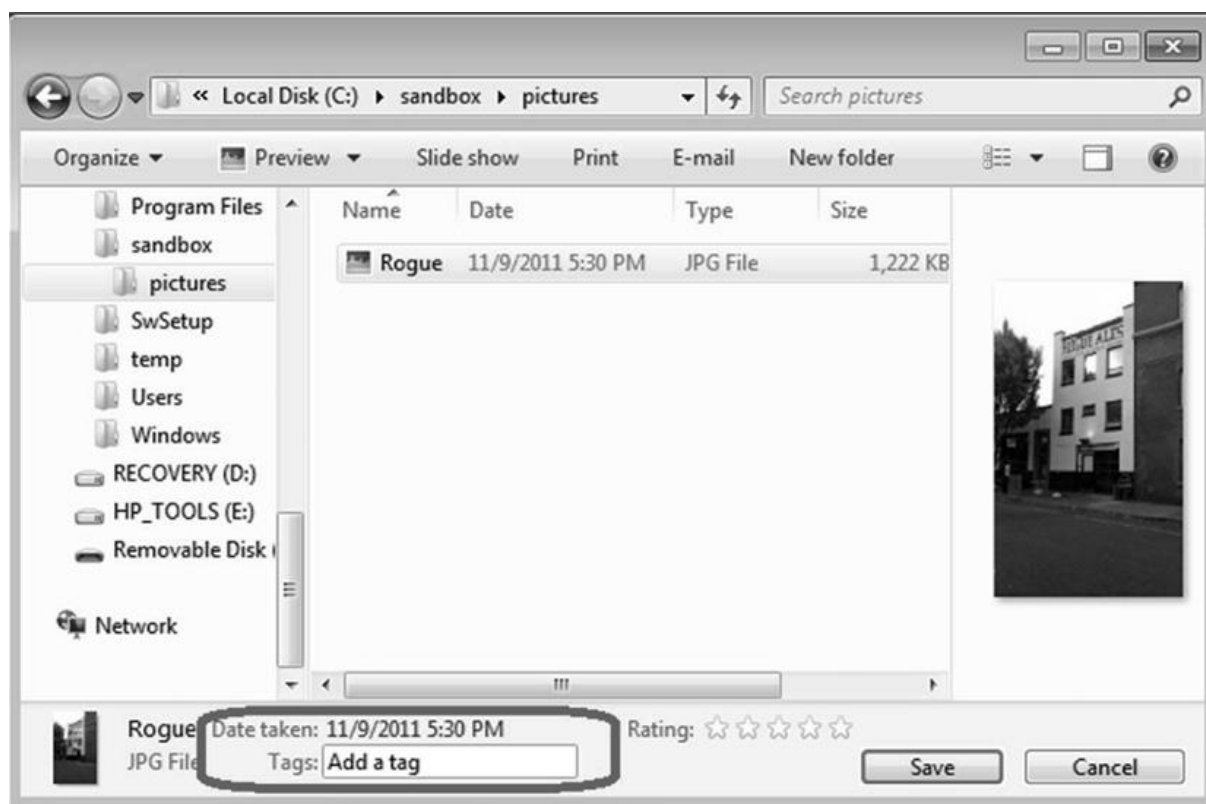
نرم افزار Picasa همچنین امکان ویرایش یکی دیگر از هدرهای استاندارد EXIF به نام برچسب<sup>۱</sup> در عکس را می دهد. همان گونه که پیش تر اشاره شد برچسب ها، مکان بدوی پنهان سازی داده ها به دور از چشمان کاربران کنجکاو و محصولات امنیتی است. برای تغییر داده های Tag کافی است گزینه SHOW/HIDE را فعال نمایید (عکس ۲-۱۰).

<sup>۱</sup> tag



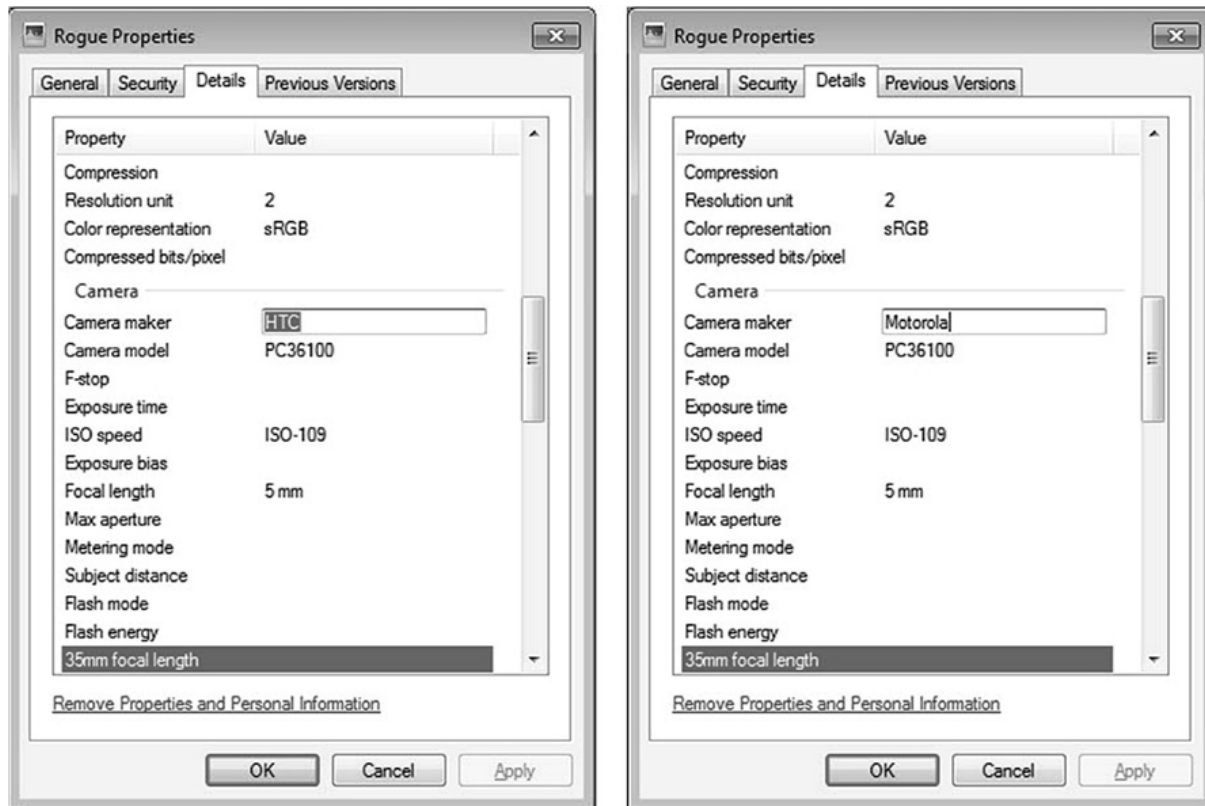
شکل ۲-۱۰: افزودن برچسب به عکس در استاندارد EXIF

ویندوز ۷ نیز راه ساده‌ای برای مشاهده و حتی تغییر داده‌های سرایند یا ابر داده‌های موجود در فایل تصویر با استاندارد Exif را ارائه می‌دهد (عکس ۲-۱۱).



شکل ۲-۱۱: مشاهده ابر داده‌های تصاویر در ویندوز ۷

ویندوز با تلیک راست بر روی نام عکس، امکان تغییر سراینده را نیز فراهم می کند. مثلاً می توان سازنده ی دوربین را از HTC به Motorola به شکل زیر تغییر داد (عکس ۲-۱۲).



شکل ۲-۱۲: تغییر نام سازنده دوربینی از HTC به Motorola

گرچه این راه ها، روش های ابتدایی برای پنهان سازی اطلاعات در عکس ها را ارائه می دهد، اما در عین حال، راه ساده ای است که بیشتر مردم توانایی انجام آن را دارند؛ در نتیجه گزینه ی در دسترس و راه حل کارایی برای پنهان کردن داده ها و ارسال آن به دیگران است. کاربران ناآگاه از این موضوع می توانند اطلاعات هویتی خود را در عکس ها به جای بگذارند و خطر انتشارش را در اینترنت بپذیرند. مراجع ذیصلاح قانونی از این ابرداده ها به ویژه داده های GPS برای ره گیری اشخاص و زمان و حتی دوربینی که عکس با آن گرفته شده است، استفاده می کنند.

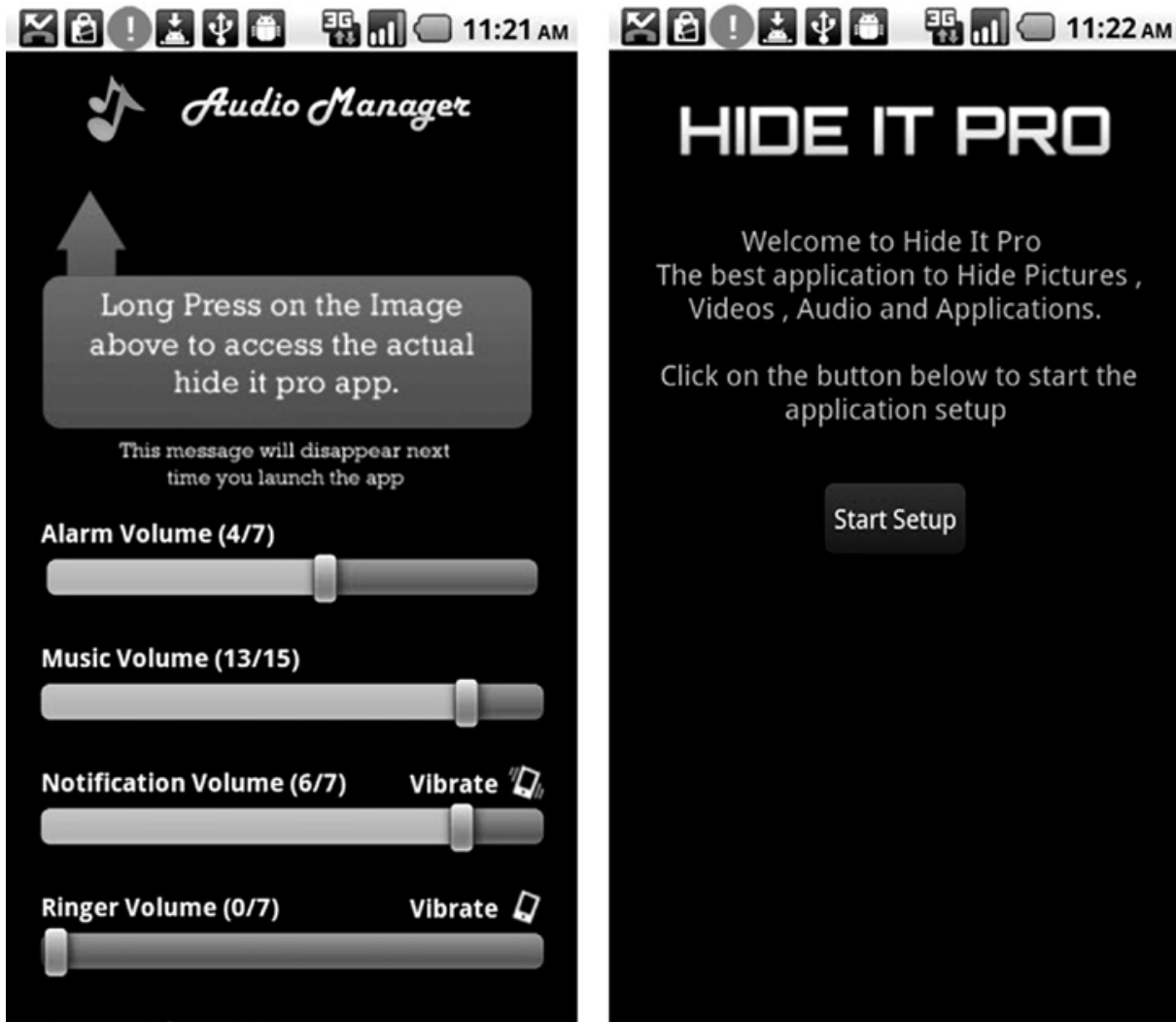
## پنهان‌سازی داده‌ها در ابزارهای همراه

سیستم‌عامل اندروید شرکت گوگل بر پایه‌ی هسته لینوکس<sup>۱</sup> بنانهاده شده و در طیف گسترده‌ای از محصولات همراه کاربرد دارد. این فراگیری کاربرد باعث زیاده‌روی در تولید نرم‌افزارهای آن شده است؛ به ویژه، شکلی از امنیت که در لینوکس پایه پیاده‌سازی شده به تولیدکنندگان اجازه می‌دهد تا نگارش ویژه‌ی خود را براساس ویژگی‌های سخت‌افزاری تولید خود، پیاده‌سازی نمایند.

برخی از نرم‌افزارها، عملکردهای ذاتی لینوکس در پنهان‌سازی داده‌ها را در اختیار کاربران قرار می‌دهند. نرم‌افزار Google play برخی از این ویژگی‌ها را برای گریز از تشخیص داده‌های پنهان به وسیله‌ی مرورگرهای دقیق فراهم می‌کند. این نرم‌افزار به گونه‌ای طراحی شده تا امکان پنهان کردن فایل‌های کاربر را از دید سایر کاربران در تلفن‌های هوشمند اندرویدی فراهم نماید.

این نرم‌افزار، خود را تحت عنوان ساختگی Audio Manager پنهان کرده و در سیستم نصب می‌شود. در زمان فراخوانی برنامه Audio Image برای چند ثانیه اجرا شد سپس نرم‌افزار Hide it pro را اجرا کرده و راه هوشمندانه‌ای را در زمینه امنیت در اختیار شما قرار می‌دهد (عکس ۲-۱۳).





شکل ۲-۱۳: نرم افزار Hid it pro

به علاوه دسترسی به خود نرم افزار توسط کلمه عبور یا PIN محافظت شده و به عنوان لایه دوم حفاظت از نرم افزار به کار می رود (عکس ۲-۱۴).



شکل ۲-۱۴: صفحه ورد به نرم‌افزار Hid it pro

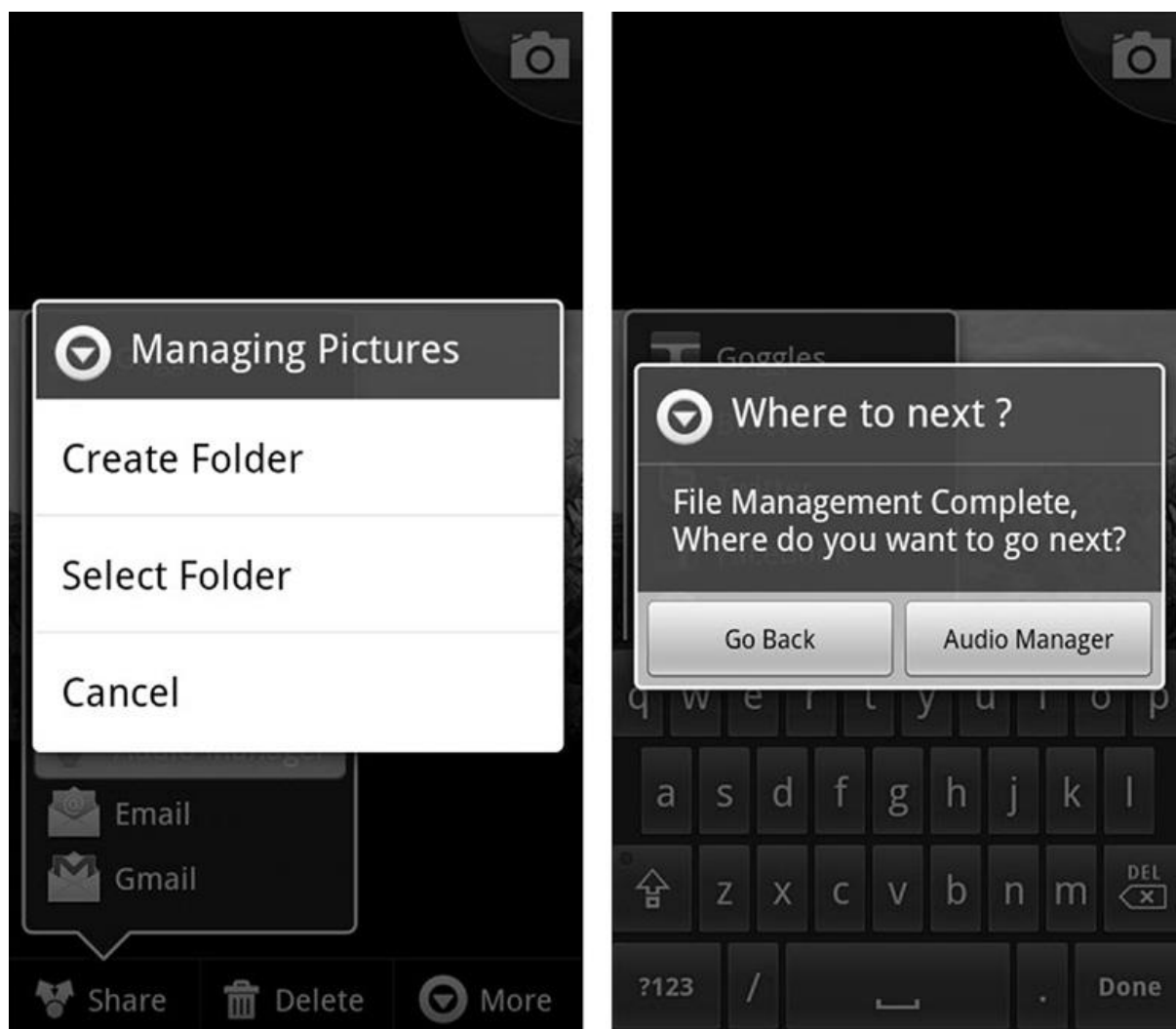
Hide it pro پوشه‌ی جدیدی را برای ذخیره فایل‌ها ایجاد کرده و پسوند دلخواه شما را به آن‌ها اختصاص می‌دهد. این پوشه بر روی حافظه‌ی جانبی SD و در شاخه‌ی ProgramData/Android/Language /mnt/sdcard ذخیره می‌شود، بنابراین، این نرم‌افزار خود در زیر شاخه language و بر روی کارت حافظه جانبی، پوشه ویژه خود را ایجاد می‌کند.

در سیستم عامل Linux پوشه ها و فایل هایی که اسم آنها با نقطه شروع می شوند، پنهان هستند. Hide it pro از این ویژگی در تغییر چهره ی خود و ذخیره ی فایل هایش بر روی SD استفاده می کند. شایان یادآوری است که Andriod از نمایش فایل هایی که اسامی آنها با نقطه شروع می شود چشم پوشی می کند. Hide it pro از این فن برای پنهان کردن فایل از دید کاربران کنجکاوی که به دنبال فایل چندرسانه ای هستند استفاده می کند. مراحل کار با این نرم افزار ساده است، به این صورت که فایل را انتخاب کرده، سپس با انتخاب گزینه Show امکان استفاده از نام ساختگی نرم افزار Audio Manager و دسترسی به Hide it pro فراهم می شود و پس از آن پنجره ی مدیریت فایل باز شده و امکان ایجاد یا انتخاب محلی برای انتقال فایل به وسیله ی Audio manager و در واقع توسط Hide it pro فراهم می شود (عکس ۲- ۱۵).



عکس ۲- ۱۵ : مشاهده عکس ها گالری و امکان استفاده از Hidit pro از آنها

سیستم مدیریت فایل، پنجره‌ای را نمایش داده مه به شما امکان ایجاد یا انتخاب پوشه و انتقال فایل را به می‌دهد (عکس ۲-۱۶).



شکل ۲-۱۶: **hid it pro** امکان انتخاب پوشه‌های موجود یا ایجاد پوشه‌ی جدید را می‌دهد

فایل‌های ذخیره شده در **Hide it pro** به فایلی با پسوند **bin** تغییر نام می‌یابند. با لیست گرفتن از پوشه‌ها در **Linux** امکان مشاهده‌ی پوشه و فایل‌های تغییر نام داده شده، همان‌گونه که در شکل ۲-۱۷ مشاهده می‌کنید، وجود دارد.

```

C:\WINDOWS\system32\cmd.exe - adb shell
.sg
.tp
.ru
$ cd .fr
$ ls
ls
Pictures
Videos
Audio
$ cd Pictures
cd Pictures
$ ls
ls
Random
nike
$ cd mike
cd mike
$ ls
ls
IMG_20110523_065614.bin
$ pwd
/mnt/sdcard/ProgramData/Android/Language/.fr/Pictures/nike
$

```

In addition to .nomedia, it creates linux hidden directories .\*

And renames files with a \*.bin extension (for example)

شکل ۲-۱۷: فایل ها و پوشه های ایجاد شده به وسیله نرم افزار Hid it pro

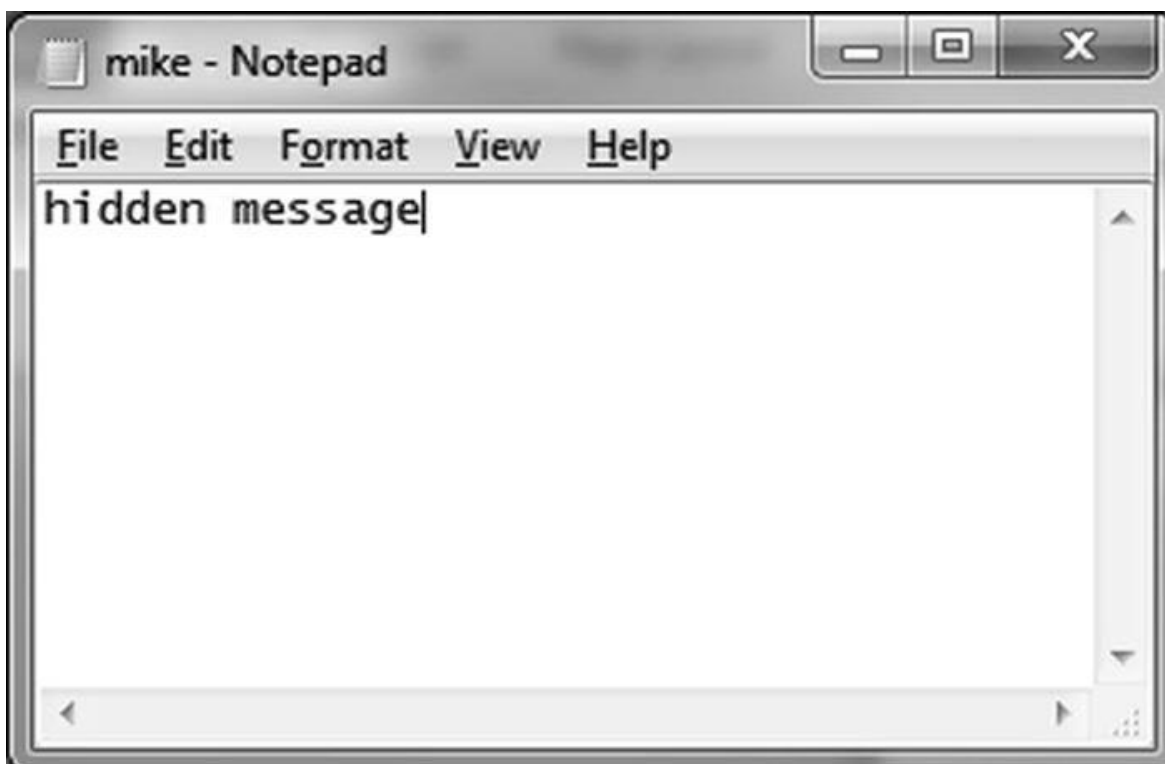
وقتی که نرم افزار Hide it pro فایل چندرسانه ای را پنهان کرد، اصل فایل از گالری اندروید حذف می شود. البته فایل پنهان همچنان به وسیله Hide it pro قابل دسترسی است؛ به این صورت که نمایه آیکون Audio Manager را به مدت ۵ ثانیه نگهدارید و گذر واژه را وارد کنید (عکس ۲-۱۸).



شکل ۲-۱۸: دسترسی به فایل های پنهان در نرم افزار Hid it pro

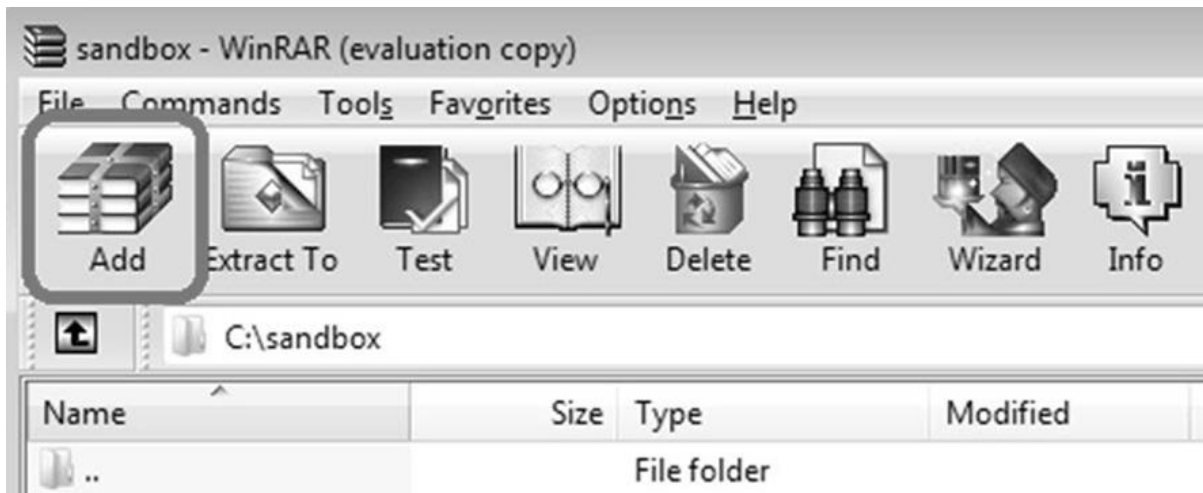
## پنهان‌سازی داده‌ها در نرم‌افزارهای فشرده‌سازی فایل

نرم‌افزار WinRAR یکی از پرکاربردترین ابزارهای فشرده‌سازی در کنار نرم‌افزار WinZip است. WinRAR به وسیله‌ی سیستم‌عامل‌های Linux, Mac OS و ویندوز مایکروسافت پشتیبانی می‌شود و نکته‌ی جالب در مورد آن این است که می‌تواند فایل‌های آرشیو آسیب دیده را ترمیم نماید. این ویژگی امکان پنهان کردن داده‌ها در فایل فشرده را نیز فراهم می‌کند. گیرنده با استفاده از ویژگی خود ترمیمی WinRAR می‌تواند اقدام به آشکار کردن فایل پنهان در فایل حامل نماید. برای شروع، فایل فشرده‌ای که می‌خواهیم فایل پنهان را در آن جاسازی کنیم، ایجاد می‌کنیم. نخست برای پنهان کردن پیام فایل، متنی که پیام را در آن تایپ کرده‌ایم را ایجاد و mike.txt می‌نامیم.

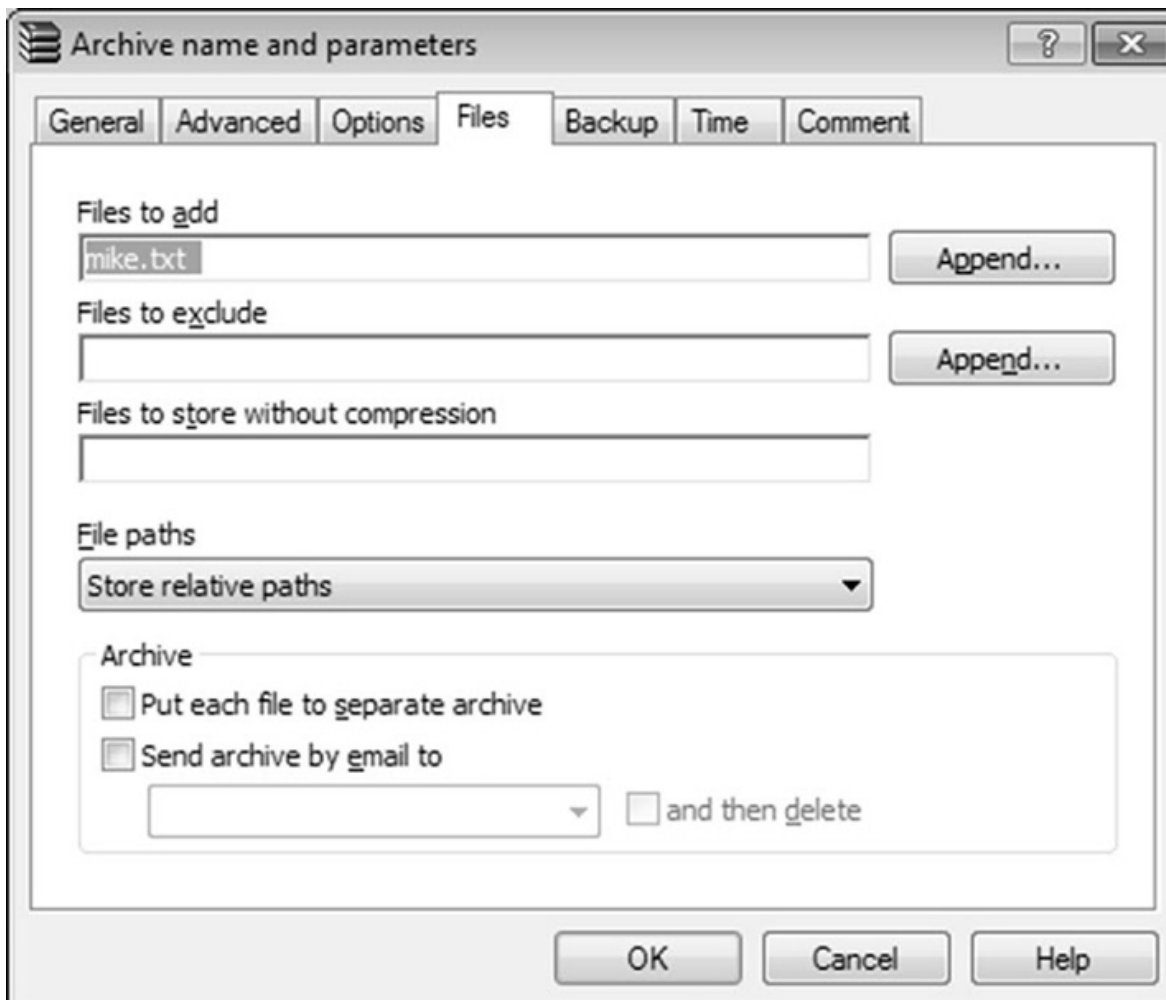


عکس ۲-۱۹: فایل متنی محتوای پیام پنهانی

سپس فایل آرشیو به نام mike.rar که حاوی فایل mike.Txt هم هست را ایجاد می‌کنیم. برای این کار پس از اجرای نرم‌افزار winRAR با استفاده از گزینه‌ی ADD، فایل متنی را که حاوی پیام پنهانی است به آرشیو اضافه می‌کنیم؛ سپس ok را کلیک کرده تا فایل آرشیو mike.rar درست شود (عکس ۲-۲۰ و ۲۱-۲).

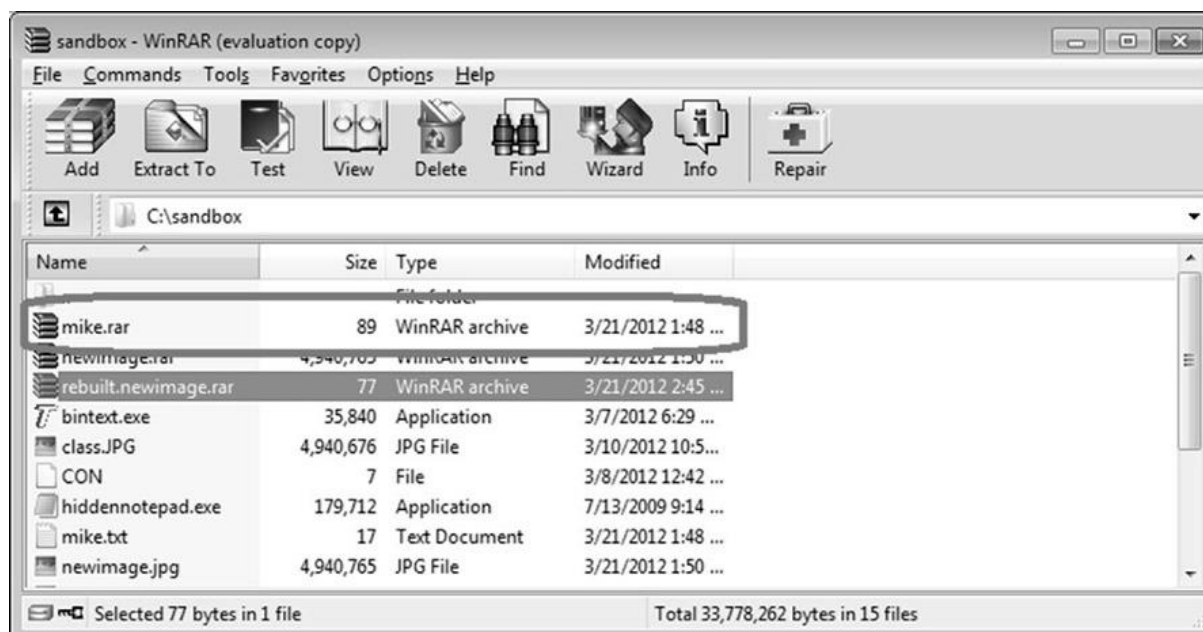


عکس ۲-۲۰: اجرای نرم افزار WinRAR و انتخاب گزینه ی ADD



عکس ۲-۲۱: افزودن فایل Mike.txt به فایل فشرده

اکنون فایل آرشیو را در فایل عکس با قالب JPEG پنهان می‌کنیم. برای این کار از فایل عکس به نام Class.jpg به عنوان فایل حامل استفاده کرده و با به کار بردن ویژگی b/ در دستور copy که امکان تغییر شکل فایل به باینری را در اختیارمان قرار می‌دهد سود می‌بریم. ضمناً از دستور + در برنامه cmd برای ترکیب دو فایل و ایجاد فایل دیگری به شکل زیر استفاده می‌کنیم (عکس ۲-۲۲).



عکس ۲-۲۲: ایجاد فایل فشرده‌ی Mike.rar

```
c:\sandbox>copy /b class.jpg + mike.rar newimage.jpg
class.JPG
mike.rar
1 file(s) copied.
```

با این روش، امکان اضافه کردن فایل آرشیو به فایل عکس و پس از نشانگر EOF موجود در عکس را داریم. ویژگی افزودن داده‌ها یکی از نشانگر EOF به فایل عکس امکان می‌دهد به صورت عادی به نمایش درآید و از داده‌های پس از علامت EOF به سادگی چشم‌پوشی می‌کند و این محل را به مکان مناسبی برای پنهان کردن داده‌ها تبدیل نماید. تا این مرحله محتوای پیام پنهان را ابتدا با WinRAR آرشیو و سپس در فایل عکس پنهان کردیم. اکنون این فایل که از ترکیب دو فایل فشرده و عکس تولید شده را می‌توان برای مقصد ارسال کرد. حال مراحلی که باید به وسیله‌ی گیرنده انجام شود را بررسی می‌کنیم.

به هنگام دریافت فایل، گیرنده پسوند فایل عکس دریافتی را از JPEG به پسوند RAR به شکل زیر تغییر می‌دهد.

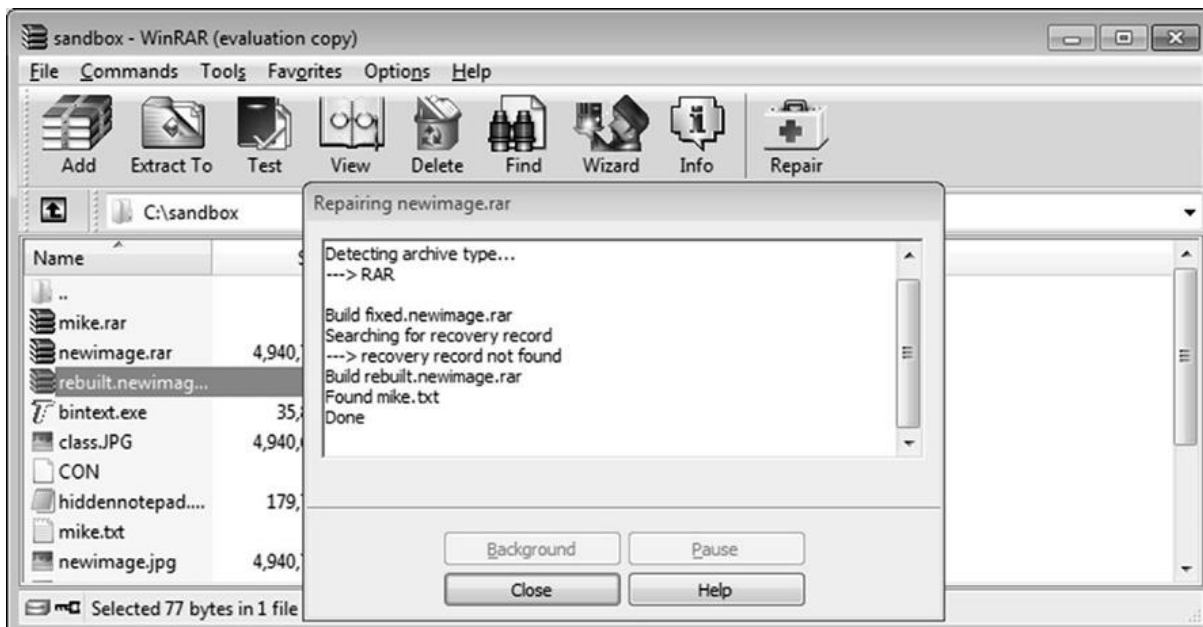


```

c:\sandbox>copy newimage.jpg newimage.rar
1 file(s) copied.
c:\sandbox>dir
Directory of C:\sandbox
04/27/2012 11:43 AM <DIR> .
04/27/2012 11:43 AM <DIR> ..
03/21/2012 01:50 PM 4,940,765 newimage.jpg
03/21/2012 01:50 PM 4,940,765 newimage.rar

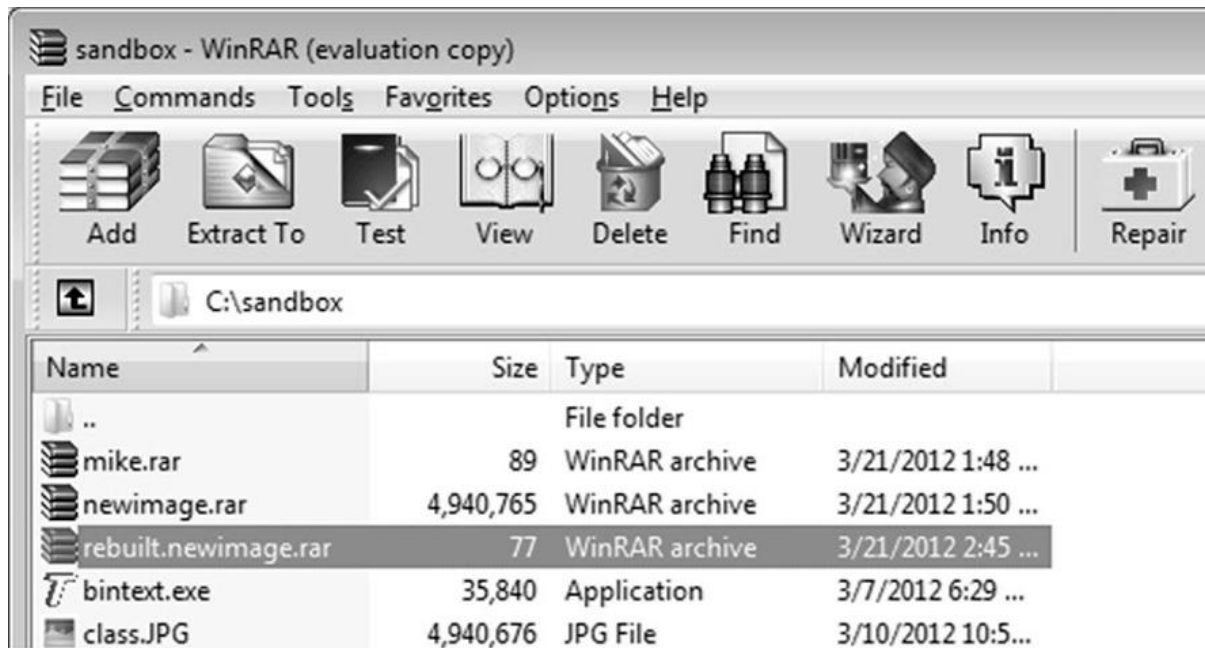
```

WinRAR امکان بازسازی فایل های فشرده آسیب دیده را ارائه می دهد. پس از این ویژگی برای آشکار کردن پیام پنهان می توان استفاده کرد و با انتخاب Repair Damaged و انتخاب فایل RAR مورد نظر، نرم افزار WinRAR، فایل jpeg حامل و فایل آرشیو همراه آن را تشخیص داده و فایل آرشیو را از فایل عکس جدا کند (عکس ۲-۲۳).



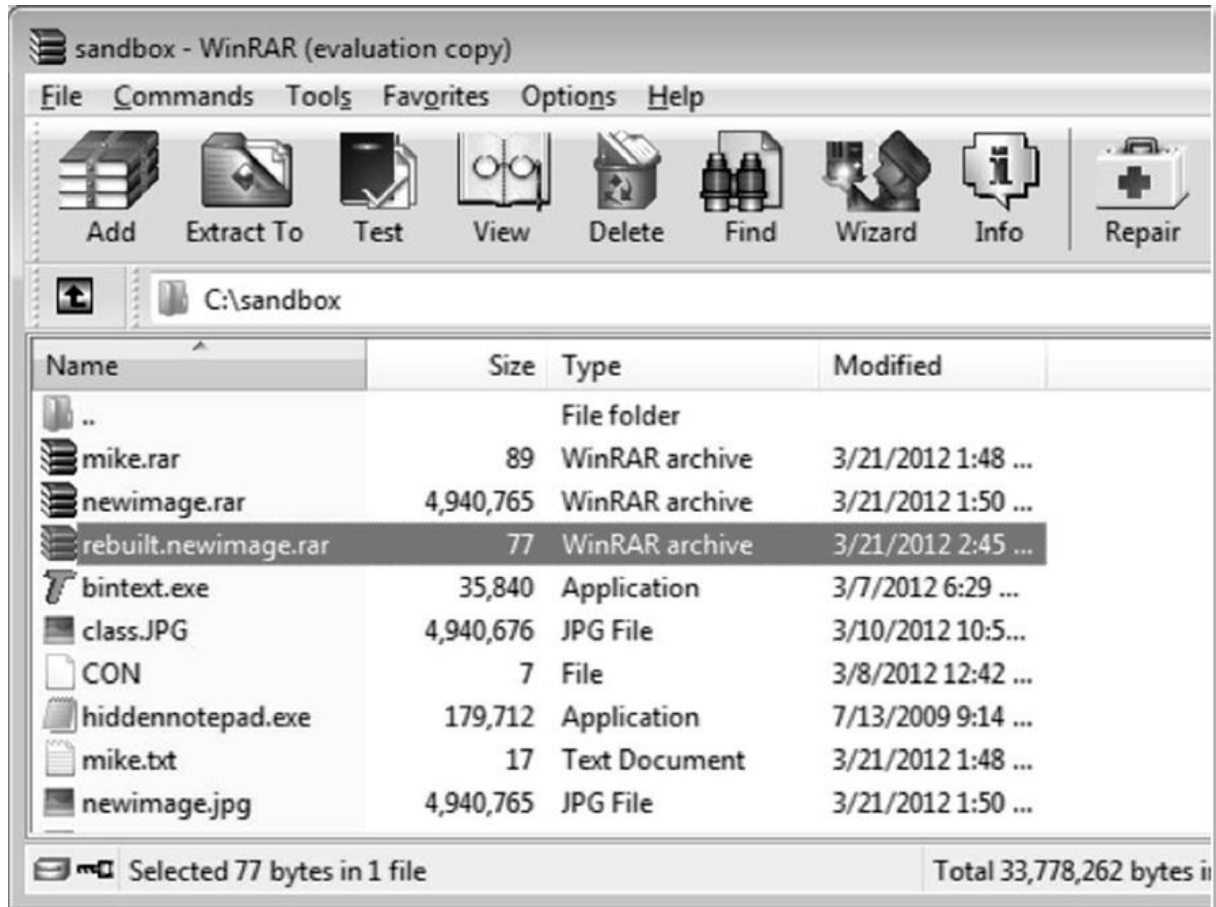
شکل ۲-۲۴: بازسازی فایل فشرده با داده های پنهان

با این کار باعث بازسازی فایل فشرده که محتوای فایل Mike.txt هم هست می شویم و فایل بازسازی شده تحت عنوان rebuilt.newimage.rar ذخیره می شود (عکس ۲-۲۴).



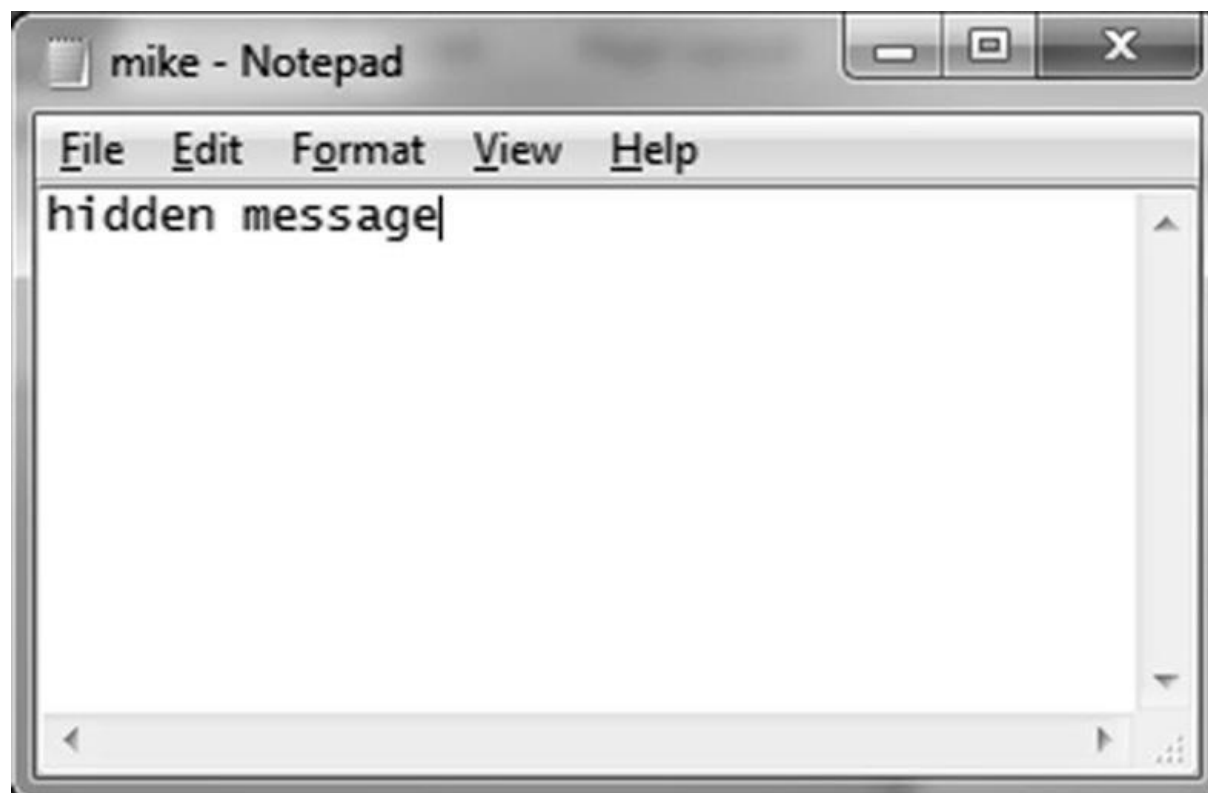
شکل ۲-۲۴: بازسازی فایل فشرده با داده‌های پنهان

اکنون با استفاده از گزینه Extract To، گیرنده قادر به باز کردن فایل متنی موجود در داخل فایل آرشیو که طی مراحل بالا ساخته شده می‌باشد و به وی اجازه مشاهده‌ی پیام متنی پنهان را می‌دهد (عکس ۲-۲۵).



شکل ۲-۲۵: با استفاده از گزینه Extract فایل آرشیو و فایل متنی همراه آشکار می شود.

گیرنده، اکنون پیام پنهان را به صورت زیر مشاهده می نماید (عکس ۲-۲۶).



شکل ۲-۲۶: آشکارسازی پیام پنهان به وسیله‌ی گیرنده.

## چکیده

در این فصل چند ابزار و روش پنهان‌سازی داده‌ها در فایل‌های پر استفاده‌ای چون ورد و ابزارهای ویرایش و ایجاد ابرداده‌ها در فایل‌ها و همچنین راه‌های گوناگون پنهان‌سازی داده‌ها در فایل را بررسی کردیم. به علاوه بسیاری از نرم‌افزارها مثل WinRAR، ویژگی پنهان‌سازی داده‌ها، بدون امکان تشخیص در نرم‌افزار مورد استفاده را فراهم می‌کنند.

همان‌گونه که در این کتاب به سوی استتار واقعی در دنیای دیجیتال پیش می‌رویم، نیاز به روش‌ها و الگوریتم‌هایی که با استفاده از آن‌ها بتوان داده‌های پنهان شده را از دید نرم‌افزارهایی که فایل حامل را نمایش می‌دهند، پنهان کرد، بیش از پیش احساس می‌شود. برای این که استتار حرفه‌ای مؤثر باشد، می‌بایست داده‌های پنهان، از دید عملیات عادی در نرم‌افزار که از فایل استفاده می‌کند ناشناخته بماند.

برای مثال، مشاهده همزمان فایل اصلی و عکس که فایل دیگری را در دل خود پنهان کرده، نباید هیچ تأثیری روی بیننده تیزبین داشته باشد. فایل‌های صوتی که حامل داده پنهان هستند، در زمان پخش باید با فایل اصلی برای شنونده غیرقابل تشخیص و یکسان باشند. الگوریتم‌های تشخیص داده‌های پنهان از روش‌های آماری زیرکانه و یا هوش مصنوعی برای تشخیص فایل ساختگی از اصلی استفاده می‌کنند. به این

شیوه و بدون استفاده از فایل اصلی و مقایسه با فایل حامل، پیدا کردن داده‌های پنهان تقریباً غیرممکن است. در پروتکل‌های شبکه اگر نگوییم غیرممکن ولی بسیار مشکل است که بدون تولید تعداد زیادی تشخیص نادرست<sup>۱</sup> بتوان داده‌های پنهان شده در بسته‌ها را تشخیص داد.

---

<sup>۱</sup> False Positive

~~~~~

### پوشیده نگاری

برخی برای این عقیده‌اند که عملیات Shady RAT، بزرگترین عملیات جاسوسی در طول تاریخ است که از سال ۲۰۰۶ تا ۲۰۱۱ میلادی ادامه داشت. هدف این عملیات، سرقت دارایی‌های فکری آژانس‌های سطح بالای دولتی و پیمانکاران مرتبط با آنها بود. مهاجمان از روش‌های پوشیده نگاری برای پنهان کردن دستورات و پیام‌های کنترلی در داخل عکس‌های دیجیتال و در صفحات وب سایت<sup>۱</sup> استفاده کردند. اما پرسش اساسی این است که حمله چگونه انجام شد؟ چگونه فایل‌ها توزیع شد؟ پاسخ به این پرسش‌ها نیازمند تحلیل دقیق‌تر عملکرد شرکت‌های McAfee و Symantec researchers می‌باشد که پس از همکاری مشترک موفق به کشف این عملیات شدند.

برای کاربران خاصی که در این آژانس‌ها یا شرکت‌های پیمانکاری دولتی کار می‌کردند، ایمیل‌هایی حاوی پیوست‌هایی در قالب فایل‌های اکسل، ورد و PDF ارسال می‌شد. این فایل‌ها به گونه‌ای نامگذاری می‌شدند که گویی حاوی اطلاعاتی مرتبط با آن سازمان یا پیمانکار هستند. سپس این کاربران بی‌خبر از همه جا، وقتی فایل را باز می‌کردند باعث می‌شدند که تروجان<sup>۲</sup> بر روی کامپیوترشان نصب شود؛ سپس این تروجان، خود را به آدرس‌های URL که در فایل عکس یا فایل HTML، بوده و حاوی دستورات کنترلی است، می‌رساند.

---

<sup>۱</sup> HTML

<sup>۲</sup> Trojan

شرکت سیمانتک<sup>۱</sup> تصاویر حاوی دستورات سری که باعث آلوده شدن این کامپیوترها می‌شد و به کامپیوتر سرور "Command Control" دست می‌یافت را مشخص کرده و نشان داده که با این روش، اطلاعات از این کامپیوترها درز پیدا کرده است. همچنین سیمانتک اعلام کرد که این دستورات، با استفاده از روش‌های پوشیده‌نگاری در فایل‌ها پنهان شده‌اند. از آنجا که اکثر فایروال‌ها و فیلترهای نرم-افزاری به فایل‌های HTML و فایل‌های عکس اجازه عبور می‌دهند، در نتیجه این فایل‌ها بدون این که شناسایی شوند دوباره به کاربران برمی‌گردد.

مهاجمان در زمینه‌ی چگونگی استفاده از روش‌های پوشیده‌نگاری بیش از پیش مهارت کسب کرده‌اند. از نگاه تاریخی، پوشیده‌نگاری برای برقراری ارتباط پنهان و پنهان کردن اطلاعات حساس استفاده شده است. نمودار شکل ۳-۱ نوآوری‌های کلیدی در سیر زمان را در عصر استتار دیجیتالی نشان می‌دهد.

### Advancements in digital steganography

|           |       |                                                             |
|-----------|-------|-------------------------------------------------------------|
| 1996      | ..... | Stools                                                      |
| 1997      | ..... | Covert TCP                                                  |
| 1998      | ..... | Camouflage                                                  |
| 1999      | ..... | JPHS                                                        |
| 2000-2004 | ..... | Copycats (Steganos, JSTEG, GIF-it-up, ...)                  |
| 2001      | ..... | Al Qaeda 9/11                                               |
| 2003      | ..... | Hydan (hide in executables)                                 |
| 2005      | ..... | InvisibleSecrets                                            |
| 2006-2011 | ..... | Shady RAT (steganography used for CnC commands)             |
| 2007-2010 | ..... | Multimedia Steganography                                    |
| 2010      | ..... | Russian Spy Case (steganography used for communications)    |
| 2011      | ..... | Alureon Trojan uses steganography                           |
| 2011      | ..... | Germans discover evidence of steganography used by Al Qaeda |
| 2012      | ..... | iOS and Android data hiding methods and applications arrive |

شکل ۳-۱: نمودار پیشرفت‌ها در استتار دیجیتالی

در پنج سال گذشته شاهد رشد استفاده از پوشیده‌نگاری در بدافزارها هم بوده‌ایم. این روش، به مهاجمان اجازه می‌دهد که علی‌رغم وجود فایروال‌ها، فیلترهای برنامه‌ی وب، سیستم‌های پیشگیری از نفوذ و سایر لایه‌های امنیتی، بدون شناسایی شدن نرم‌افزارهای مخرب را بگسترانند. اجازه دهید نگاهی

<sup>۱</sup> Symantec



به شیوه‌های اساسی استتار داده‌ها در فایل‌های حامل، چون عکس‌های دیجیتال، صفحات HTML و سایر انواع فایل‌های رایج داشته باشیم.

## شیوه‌های پوشیده‌نگاری

همانگونه که در فصل اول اشاره شد، استتار به معنی نوشتن در لفافه و یا نوشتن نامرئی است. در استتار دیجیتال، کاربر معمولاً از نرم‌افزاری برای پنهان کردن پیام یا فایلی در فایل حامل استفاده می‌کند؛ سپس اقدام به ارسال فایل حامل به گیرنده یا پست کردن آن به سایتی برای دانلود به وسیله‌ی گیرنده می‌نماید. گیرنده پس از دریافت فایل از همان نرم‌افزار برای آشکار کردن پیام یا فایل پنهان استفاده می‌کند. برخی نرم‌افزارها، برای حفاظت بیشتر از پیام پنهان، از گذرواژه هم استفاده می‌کنند و برخی دیگر پیام را رمزنگاری کرده، سپس با استفاده از گذرواژه از محتوای پنهان شده حفاظت می‌کنند. روش‌ها و تکنیک‌های گوناگونی که برای پنهان‌سازی داده‌های دیجیتال وجود دارد به دو گروه اصلی تقسیم می‌شود:

(۱) درج<sup>۱</sup>: در این روش داده‌هایی را که می‌خواهیم پنهان کنیم، به داده‌های اصلی فایل اضافه می‌شود و این داده‌ها می‌تواند شامل متن پنهان یا نشانگرهای از فایل اصلی باشند که به عنوان راهنمای نرم‌افزار برای تشخیص محل داده‌های پنهان در فایل اصلی مورد استفاده باشد. معمولاً این روش، از فضای خالی موجود در فرمت فایل‌ها استفاده می‌کند.

(۲) جایگزینی<sup>۲</sup>: جایگزینی عبارت است از تغییر یا تعویض موقعیت بایت‌های موجود در فایل اصلی به گونه‌ای که چیز جدیدی به فایل حامل اضافه نشود، بلکه فقط بایت‌های موجود به گونه‌ای تغییر کند که داده‌های پنهان را نشان داده و تغییرهای ایجاد شده، به صورت دیداری و شنیداری قابل تشخیص نباشد. یک نمونه از این شیوه، پوشیده‌نگاری در کم ارزش‌ترین بیت<sup>۳</sup> است. در این روش، کم ارزش‌ترین بیت یک سری بایت موجود در فایل اصلی را با بایت‌های داده‌های پنهان جایگزین می‌کنیم. این جایگزینی گاهی موجب می‌شود که بیت‌ها از ۰ به ۱ یا از ۱ به ۰ تغییر کنند.

## روش درج

<sup>۱</sup> Insertion

<sup>۲</sup> Substitution

<sup>۳</sup> Least Significant Bit

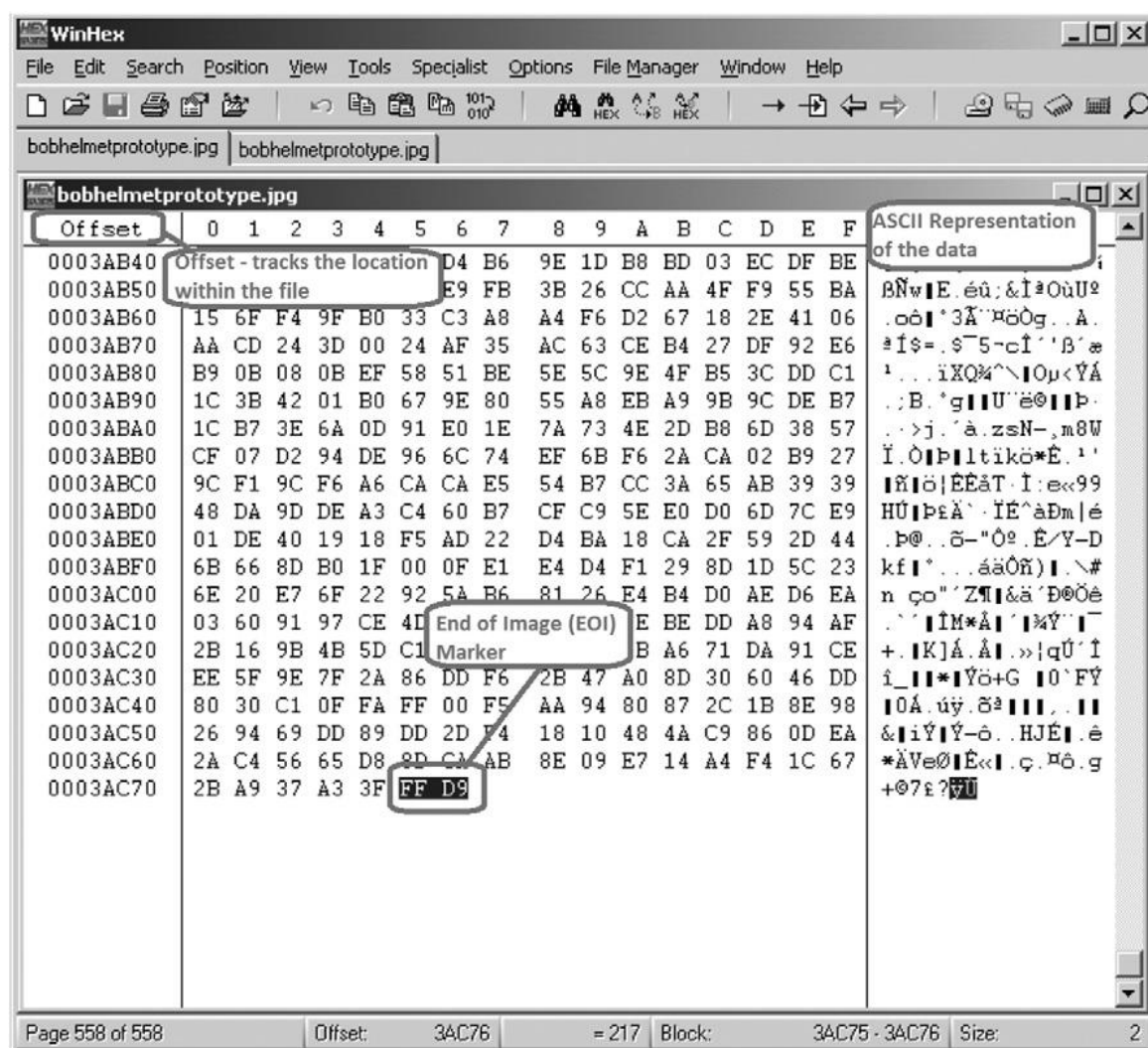
درج را می‌توان شیوه‌ای از تغییر بیت‌ها تلقی نمود؛ اما از دیدگاه پوشیده‌نگاری، درک تفاوت این دو مهم است. در پوشیده‌نگاری آنچه اهمیت دارد این است که داده‌های موجود در فایل‌های حامل دست نخورده به نظر برسند. در روش درج، داده‌های موجود در فایل تغییر نمی‌کند، بلکه داده‌های دیگری به فایل افزوده می‌شود. در روش جایگزین، داده‌های موجود در فایل بدون این که داده‌ای به آن‌ها اضافه شود، تغییر می‌کنند. از دیدگاه اندازه‌ی فایل، در هر دو حالت اندازه‌ی فایل تغییر می‌کند، اما از دیدگاه داده‌های فایل، در روش درج، داده‌هایی به اطلاعات اصلی افزوده می‌شود، در حالی که در روش تغییر، داده‌های موجود عوض می‌شود. اما همانگونه که بعداً خواهیم دید، در بسیاری از نرم‌افزارهای پوشیده‌نگاری، از ترکیبی از هر دو روش استفاده می‌شود.

### درج به روش افزودن داده‌ها به فایل

افزودن داده به انتهای فایل شاید رایج‌ترین و ساده‌ترین شکل استتار دیجیتال باشد. در بسیاری از فایل‌ها این امکان وجود دارد که بدون خراب شدن اطلاعات اصلی فایل، داده‌هایی به انتهای آن اضافه شود. شکل ۲-۳ فایل تغییر نیافته به فرمت JPEG را در نرم‌افزار WinHex نشان می‌دهد. WinHex ویرایشگر مبنای شانزده<sup>۱</sup> است و برخلاف واژه پردازها، محتویات فایل را به شکل خام نشان داده و تمام داده‌ها شامل CR و حتی کدهای اجرایی را هم نمایش می‌دهد. تمام داده‌ها در ستون وسط پنجره نرم‌افزار در مبنای شانزده و در دو رقم نشان داده می‌شوند. ستون سمت چپ، شمارنده یا افسست است که امکان دسترسی به محتویات فایل را میسر می‌کند. ستون سمت راست، داده‌ها را در قالب ASCII نشان می‌دهد. بخاطر محدودیت‌های استاندارد ASCII، تمام داده‌ها امکان نمایش در قالب ASCII را ندارند. فایل‌های JPEG یک نشانگر EOL دارد که با کدهای OxFF و OxD9 مشخص می‌شوند. همان‌گونه که در شکل ۲-۳ نشان داده شده است، می‌توانیم نشانگر EOL را در آخر فایل مشاهده کنیم.

---

<sup>۱</sup> hex

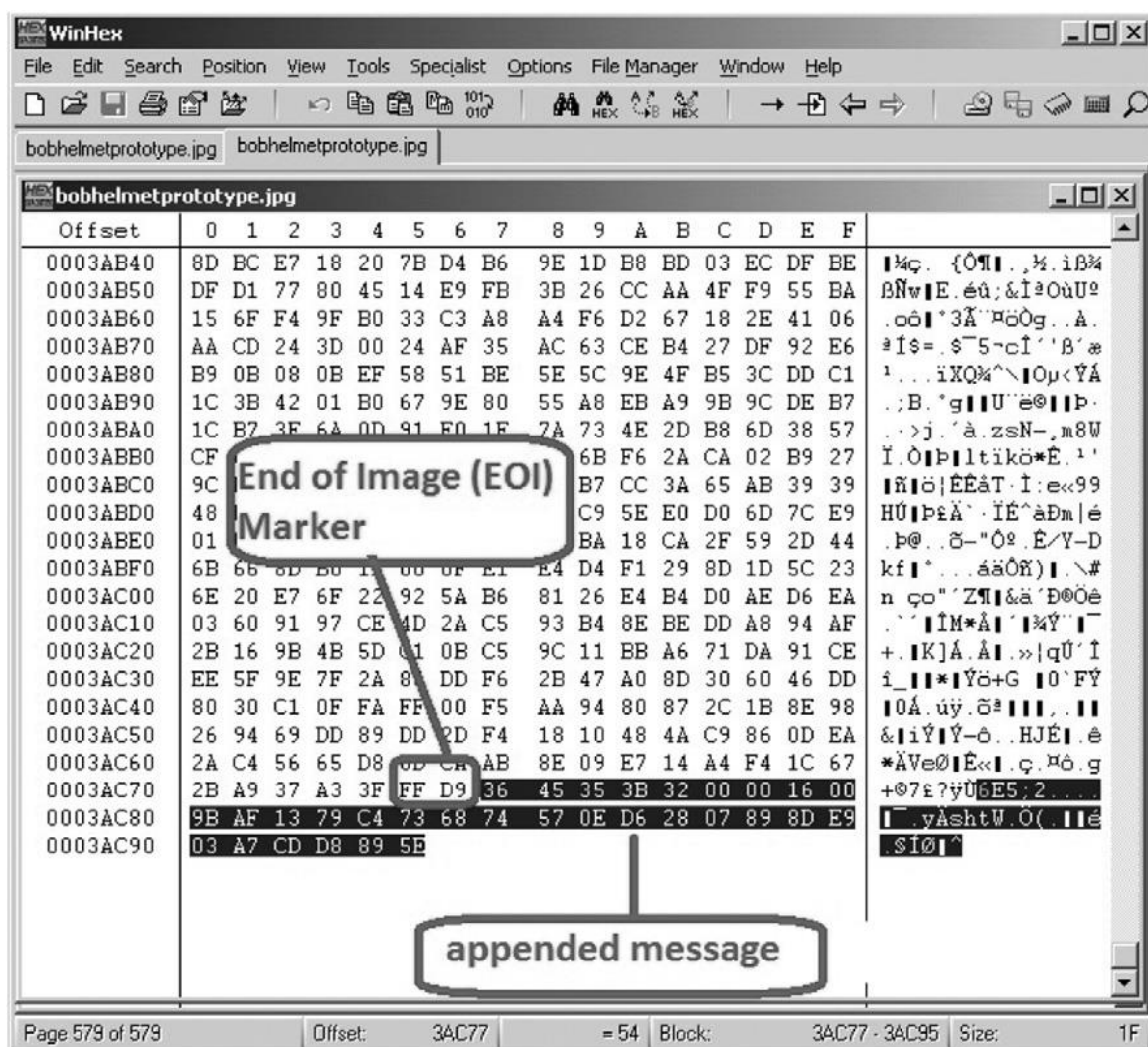


شکل ۳-۲: نمایش فایل JPEG تغییر یافته در نرم‌افزار WinHex

برنامه‌های پوشیده‌نگاری از این نشانگر برای پنهان کردن داده‌ها در فایل‌های JPEG استفاده می‌نمایند. همانگونه که در شکل ۳-۳ نشان داده شده است، می‌توانیم برای مشاهده‌ی داده‌های افزوده شده به انتهای فایل، آن را در WinHex بررسی کرد. توجه داشته باشید که این داده‌ها، پس از نشانگر EOI و 0xFF و xD90 به فایل حامل افزوده می‌شود. معمولاً داده‌های پنهان شده پس از نشانگر EOI در هنگام مشاهده، معمولی فایل نادیده گرفته می‌شوند، اما همیشه نشانه‌هایی از فایل تغییر یافته با داده‌های پنهان وجود دارد.

## درج به شیوه Prepend

هر فایلی که فیلدی برای توضیحات داشته باشد، امکان افزودن داده‌های پنهان به فایل را بدون تأثیر بر ویژگی‌های دیداری آن فراهم می‌نماید. برای مثال، فایل‌های Html و JPEG در استفاده از این تکنیک بسیار مستعدند. در فایل‌های JPEG حداکثر تا ۶۵ بایت توضیح را می‌توان به فایل عکس اضافه کرد، بدون این که بر ویژگی‌های دیداری آن تأثیری بگذارد. فایل‌های JPEG با استفاده از نشانگرها، به چند بخش تقسیم می‌شوند و هر بخش، با نشانگر OXFF مشخص می‌گردد. هر بخش داده‌های جداگانه‌ای مربوط به چگونگی نمایش تصویر مثل قالب نمایش رنگ تصویر و جزئیات دیگری دارد که در جدول ۳-۱ به آنها اشاره شده است.



شکل ۳-۳: فایل عکس در قالب Jpeg با داده‌های پنهان شده در پایان فایل

جدول ۳-۱: قالب فایل عکس Jpeg

| Table 3.1 JFIF (JPEG File Image Format) |             |              |                                      |
|-----------------------------------------|-------------|--------------|--------------------------------------|
| Marker                                  | Value (Hex) | Size (bytes) | Details                              |
| SOI                                     | FF D8       | 2            | Start of Image                       |
| APP0                                    | FF E0       | 2            | App Marker (file details)            |
| SOF0                                    | FF C0       | 2            | Start of Frame (width, height, etc.) |
| SOS                                     | FF DA       | 2            | Start of Scan (image itself)         |
| EOI                                     | FF D9       | 2            | End of Image/End of File (EOF)       |

با وجود فیلدهای زیاد داده‌ای در قالب فایل JPEG، مکان‌های زیادی در دسترس‌اند که امکان دارد در آن‌ها داده‌هایی پنهان شده باشد. شکل ۳-۴ تفاوت فایل تغییر یافته و فایل بدون تغییر را نشان می‌دهد. به داده‌های درج شده و تغییر داده‌ها بین نشانگرهای 0xFF 0xC0 و 0xFF 0xE0 توجه کنید.

| Offset   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | A  | B  | C  | D  | E  | F  |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 00000000 | FF | D8 | FF | E0 | 00 | 10 | 4A | 46 | 49 | 46 | 00 | 01 | 02 | 01 | 00 | 90 |
| 00000001 | 00 | 00 | FF | E0 | 06 | 6D | 4A | 46 | 58 | 58 | 00 | 10 | FF | D8 |    |    |
| 00000020 | FF | DB | 00 | 43 | 00 | 0A | 07 | 07 | 08 | 07 | 06 | 0A | 08 | 08 | 08 | 0E |
| 00000030 | 0A | 0A | 0B | 0E | 18 | 10 | 0E | 0D | 0D | 0E | 1D | 15 | 16 | 11 | 18 | 23 |
| 00000040 | 1F | 25 | 24 | 22 | 1F | 22 | 21 | 26 | 2B | 37 | 2F | 26 | 29 | 34 | 29 | 21 |
| 00000050 | 22 | 30 | 41 | 31 | 34 | 39 | 3B | 3E | 3E | 3E | 25 | 2E | 44 | 49 | 43 | 3C |
| 00000060 | 48 | 37 | 3D | 3E | 3B | FF | DB | 00 | 43 | 01 | 0A | 0B | 0B | 0E | 0D | 0E |
| 00000070 | 1C | 10 | 10 | 1C | 3B | 28 | 22 | 28 | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B |
| 00000080 | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B |
| 00000090 | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B |
| 000000A0 | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | 3B | FF | C0 |    |    |    |    |
| 000000B0 | 3C | 00 | 50 | 03 | 01 | 21 | 00 | 02 | 11 | 01 | 03 | 11 | 01 | FF | C0 |    |

شکل ۳-۴: Prepending داده‌ها برای فایل JPEG

فیلد توضیحات در فایل JPEG، امکان پنهان‌سازی مقدار قابل توجهی داده با کمترین تغییر در فایل اصلی را فراهم می‌کند. اگر چه فیلد توضیح در فایل JPEG حداکثر می‌تواند ۶۵۵۳۳ بایت را در خود ذخیره کند، ولی حداقل مقدار آن ۲ بایت است. در مورد نشانگر APP0، ابردادها به وسیله‌ی برنامه‌ی نمایش عکس تشخیص داده نشده و نادیده گرفته می‌شود و این امر باعث می‌شود که این مکان، محل مناسبی برای پنهان کردن داده‌ها باشد.

## پنهان‌سازی به شیوه‌ی تغییر داده‌های فایل





رایج‌ترین شکل تغییر در پوشیده‌نگاری، تغییر کم ارزش‌ترین بیت از یک یا چند بایت داده‌ی فایل می‌باشد. در نتیجه‌ی پنهان‌سازی داده‌ها، برخی از یک‌ها به صفر یا از صفر به یک تغییر می‌یابد و بدین ترتیب فایل با داده‌های اصلی تغییر یافته که همان داده‌های پنهان است حاصل می‌شود. از پشت سرهم

قراردادن بیت‌های تغییر یافته در فایل حامل می‌توان پیام پنهان شده را بازسازی نمود. می‌توان ادعا کرد که تشخیص دگرگونی حاصل از تغییر کم‌ارزش‌ترین بیت به شکل دیداری- شنیداری، برای انسان تقریباً غیر ممکن است.

## کم ارزش‌ترین بیت





روش پوشیده‌نگاری با تغییر کم ارزش‌ترین بیت داده‌ها، از ویژگی نمایش رنگ‌ها در قالب ۲۴ بیتی استفاده می‌کند. در این قالب نمایش رنگ، سه رنگ اصلی قرمز، آبی و سبز وجود دارد که دقیقاً شبیه نمایش تصاویر ویدئویی در تلویزیون است که به وسیله‌ی سه کابل قرمز، آبی و سبز، سیگنال‌ها را از ویدیو به تلویزیون منتقل می‌کند.

همانگونه که در شکل ۳-۵ نشان داده شده است، در نمایش تصویر در قالب ۲۴ بیتی، به هریک از رنگ‌های قرمز، سبز و آبی، ۸ بیت جداگانه اختصاص داده می‌شود. این تعداد از بیت‌ها توانایی نمایش ۲۵۶ طیف رنگ قرمز، ۲۵۶ طیف رنگ آبی و ۲۵۶ طیف رنگ سبز را دارند.

| • Color |                                                                                     | Binary   | Decimal |   |                                                                                       |
|---------|-------------------------------------------------------------------------------------|----------|---------|---|---------------------------------------------------------------------------------------|
| • Red   |  | 11111111 | 255     | } |  |
| • Green |  | 00000000 | 0       |   |                                                                                       |
| • Blue  |  | 00000000 | 0       |   |                                                                                       |
|         |                                                                                     |          |         |   | Red                                                                                   |

شکل ۳-۵: جدول رنگ ۲۴ بیتی Platte

از آنجایی که چشم ما رنگ‌های قرمز، سبز و آبی را تشخیص می‌دهد، با ترکیب این سه رنگ در ۲۴ بیت، می‌توان رنگ هر پیکسل از تصویر را ساخت. برای مشاهده هر نقطه از عکس، یک عدد سه‌گانه در مبنای ۱۶ داریم که بیانگر رنگ‌های قرمز، آبی و سبز آن نقطه است. بسته به مقدار داده‌ای که می‌خواهیم پنهان کنیم، کم ارزش‌ترین بیت هر رنگ از صفر به یک یا از یک به صفر تغییر کرده و یا بدون تغییر باقی می‌ماند (شکل ۳-۶).

| • Color |                                                                                   | Binary   | Decimal |   |                                                                                     |
|---------|-----------------------------------------------------------------------------------|----------|---------|---|-------------------------------------------------------------------------------------|
| • Red   |  | 11111110 | 254     | } |  |
| • Green |  | 00000000 | 0       |   |                                                                                     |
| • Blue  |  | 00000000 | 0       |   |                                                                                     |
|         |                                                                                   |          |         |   | Red                                                                                 |

شکل ۳-۶: کم‌ارزش‌ترین بیت

کم‌ارزش‌ترین بیت در بایت‌های پشت سرهم با هم ترکیب شده و پیام پنهان شده را شکل می‌دهند. برای پنهان‌سازی متن در این فایل و تولید ۸ بیتی مورد نیاز یک نویسه اسکی، بایت‌های متوالی کم‌ارزش‌ترین بیت‌ها را پشت سرهم قرار می‌دهیم تا حرف موردنظر را تولید نماید (شکل ۳-۷).

| Decimal  |   | LSBs     | Hex  | ASCII |
|----------|---|----------|------|-------|
| 11111110 | } | 01100001 | = 61 | = A   |
| 00000001 |   |          |      |       |
| 00000001 |   |          |      |       |
| 11111110 |   |          |      |       |
| 00000000 |   |          |      |       |
| 00000000 |   |          |      |       |
| 00000000 |   |          |      |       |
| 00000001 |   |          |      |       |

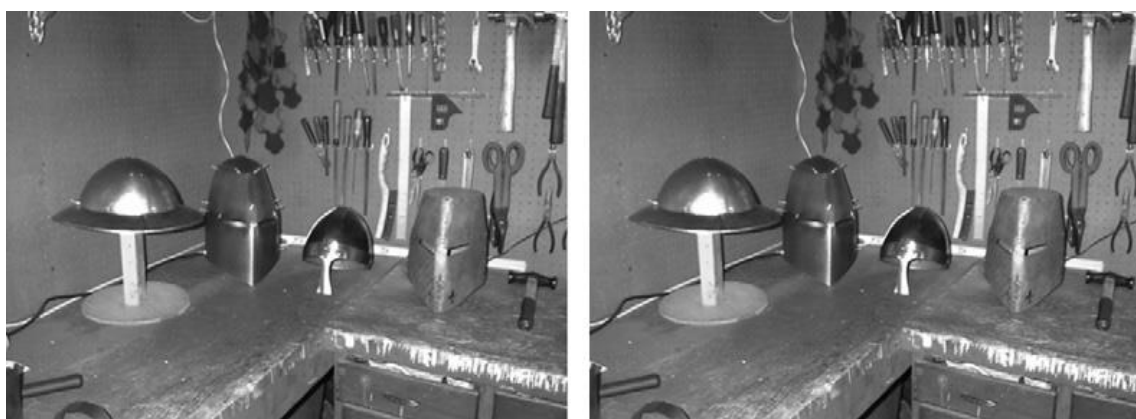
شکل ۳-۷: نمونه‌ای از پوشیده‌نگاری به روش تغییر کم‌ارزش‌ترین بیت

این شیوه‌ی پوشیده‌نگاری، راهی مناسب برای پنهان کردن داده‌هایی است که به وسیله‌ی روش‌های دیداری معمولی قابل شناسایی نباشند، به همین دلیل می‌تواند روش مطمئنی تلقی شود. این پوشیده‌نگاری را معمولاً به وسیله‌ی روش‌های تحلیل آماری می‌توان شناسایی کرد. در شکل ۳-۸ رنگ‌های به دست آمده از شکل ۳-۵ و ۳-۶ را مقایسه کنید. تشخیص تفاوت بین مقدار کم‌ارزش‌ترین بیت اصلی و تغییر یافته عملاً برای چشم غیرمسلح غیر ممکن است.



شکل ۳-۸: مقایسه نمایش دو رنگ قرمز پیش و پس از تغییر مقدار کم‌ارزش‌ترین بیت آن

نمونه‌ی پوشیده‌نگاری در شکل ۳-۹، تصویری را پیش و پس از تغییر مقدار کم‌ارزش‌ترین بیت داده‌هایش نشان می‌دهد. می‌توان ادعا کرد که شناسایی تفاوت بین دو تصویر، تقریباً غیرممکن است.



شکل ۳-۹: مقایسه فایل اصلی و فایل با تغییر یافته LSB

پراکندگی تغییرات کم‌ارزش‌ترین بیت در سراسر فایل، می‌تواند پوشش مناسبی برای ذخیره پیام پنهانی باشد و این درحالی است که ممکن است حتی اندازه‌ی فایل هم تغییر نکرده باشد. با مقایسه فایل اصلی و فایل تغییر یافته می‌توان تغییرات را تشخیص داد، اما فرض براین است که فایل اصلی و دست نخورده نزد ارسال کننده باقی مانده و ارسال نشده است.

این شیوه از پوشیده‌نگاری، در فایل‌های تصویری در قالب ۲۴ بیتی، مانند JPEG و BMP کار می‌کند. این نوع از قالب‌های فایلی را True Color نیز می‌نامند. روش تغییر کم‌ارزش‌ترین بیت بر روی فایل‌های تصویری BMP با قالب ۸ بیتی نیز به خوبی کار می‌کند. برخی نرم‌افزارهایی که این روش را در استتار به کار می‌برند عبارتند از:

S-Tools  
Image Hide  
Steganos



## پنهان‌سازی داده‌ها در فایل‌های PDF

نرم‌افزار **wbStego4open6** که آن را از آدرس <http://wbstego.wbailer.com> می‌توانید دانلود نمایید، ابزار رایگانی است که در سیستم عامل ویندوز و لینوکس کار می‌کند. این نرم‌افزار امکان پنهان نمودن داده‌ها را بدون هیچگونه تغییر قابل مشاهده‌ای در فایل‌های HTML.PDF و TXT فراهم می‌سازد. همچنین، شما می‌توانید داده‌های مربوط به کپی‌رایت<sup>۱</sup> را در این فایل‌ها جاسازی و پنهان نمایید. تعداد انگشت شماری نرم‌افزار، امکان پنهان‌سازی داده‌ها در فایل‌های PDF را می‌دهد. اجازه دهید به بررسی مراحل پنهان‌سازی داده‌ها در فایل PDF بپردازیم (شکل ۳-۱۰).



شکل ۳-۱۰: پنجره ویزارد نرم‌افزار wbStego4open

این برنامه برای افزودن اطلاعاتی که در هنگام نمایش فایل PDF در نرم‌افزار Adobe Acrobat Reader نامرتب به نظر می‌آید، از فیلد header موجود در قالب PDF استفاده می‌کند (شکل ۳-۱۱)؛ به‌علاوه، وقتی که نرم‌افزار wbStego4open داده‌ها را در فایل درج می‌کند (در مورد مثال ما، داده‌های

<sup>۱</sup> Copyright

رمزنگاری شده مربوط به کپی رایت) هم شیوه درج و هم روش تغییر کم ارزش‌ترین بیت را به کار می‌برد.

شکل ۳-۱۱: وارد کردن اطلاعات کپی رایت با استفاده از نرم‌افزار **wbStego4open**

این نرم‌افزار، کار خود را با تبدیل کد ASCII داده‌هایی که می‌خواهیم پنهان کنیم به شکل باینری شروع می‌کند. **wbStego4open** هر عدد دودویی را به شکل مبنای شانزده به شکل ۲۰ یا ۰۹ نشان می‌دهد؛ بدین ترتیب که عدد ۲۰ بیانگر صفر باینری و عدد ۰۹ بیانگر یک باینری است. به عنوان مثال، در **wbStego4open Copyright Manager**، آدرسی را وارد می‌کنیم که دارای کلمه "Oblivion" باشد. **wbStego4open** کد کاراکترهای اسکی را به معادل‌های دودویی آن‌ها تبدیل می‌کند و هر رقم دودویی را به شکل ۰x20 یا ۰x09 نشان می‌دهد (شکل ۳-۱۲).

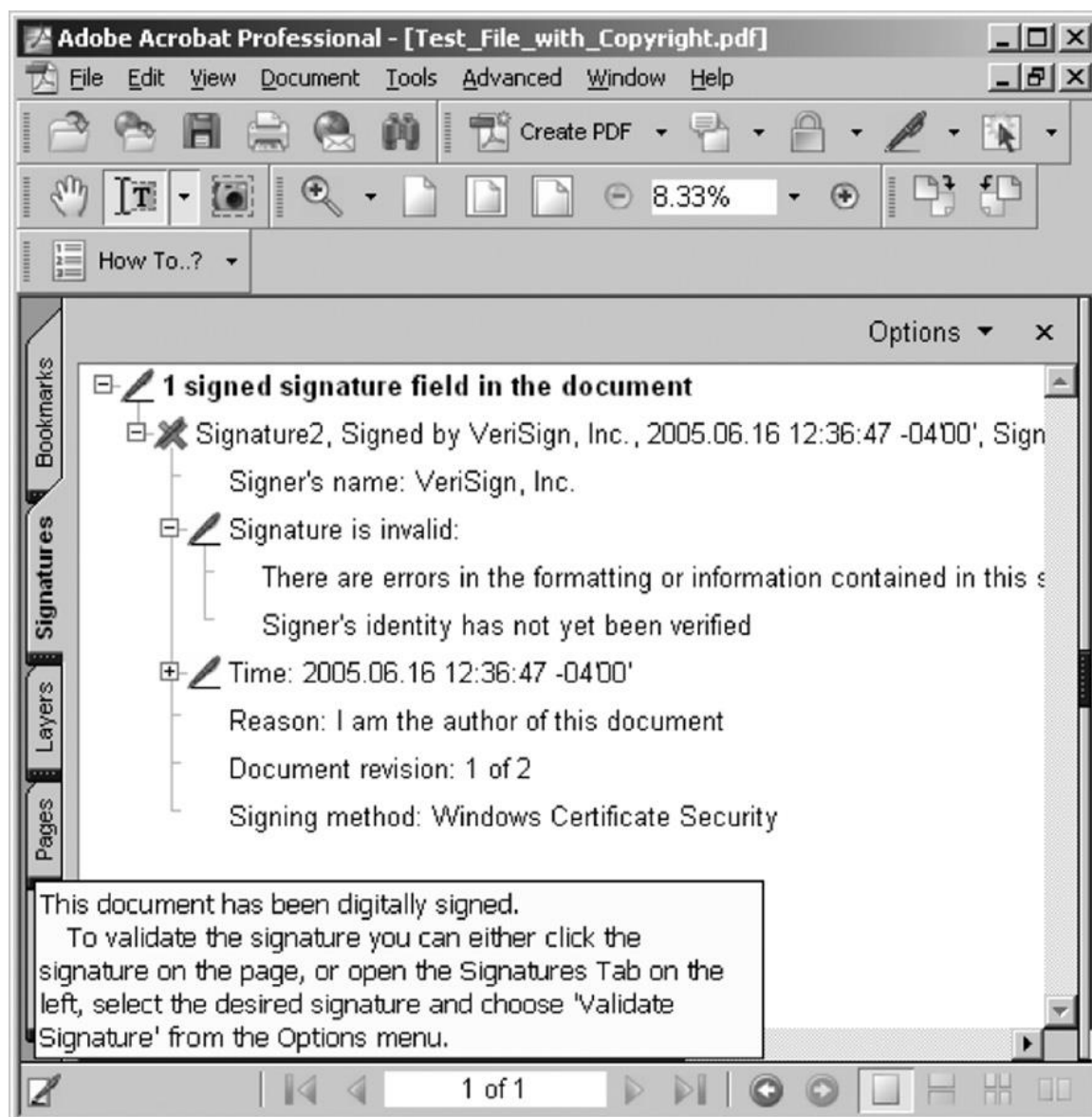
| ASCII | Binary     | Hex                     |
|-------|------------|-------------------------|
| O     | 01001111   | 20 09 20 20 09 09 09 09 |
| b     | 01100010   | 20 09 09 20 20 20 09 20 |
| l     | 01101100   | 20 09 09 20 09 09 20 20 |
| i =   | 01101001 = | 20 09 09 20 09 20 20 09 |
| v     | 01110110   | 20 09 09 09 20 09 09 20 |
| i     | 01101001   | 20 09 09 20 09 20 20 09 |
| o     | 01101111   | 20 09 09 20 09 09 09 09 |
| n     | 01101110   | 20 09 09 20 09 09 09 20 |

شکل ۳-۱۲: تبدیل ارقام باینری به اعداد در مبنای شانزده در **wbStego4open**

سپس تمام این اعداد در مبنای شانزده در فایل PDF جاسازی می‌شود. با پویش فایل به وسیله نرم‌افزار **wbStego4open** و بررسی چگونگی تغییر محتوای آن با این داده‌ها، به فایلی با داده‌هایی که شامل تعداد **ox20** و **ox09** می‌رسیم (شکل ۳-۱۳).



هرچیز، مانع کپی برداری و یا پرینت محتوای PDF می‌شود. با امضاء دیجیتال می‌تواند مانع از تغییر محتوی فایل شد. این گواهی اگرچه ممکن است اجازه تغییر در فایل را بدهد، اما گیرنده از این امر آگاه شده و بنابراین به فایل دریافتی اعتماد نمی‌کند (شکل ۳-۱۴).



شکل ۳-۱۴: نتایج تغییر یک فایل Adobe PDF دارای امضای دیجیتالی

## پنهان‌سازی داده‌ها در فایل‌های اجرایی

نرم‌افزار Hydan به آدرس (<http://www.crazyboy.com/hydan>)، ابزاری است برای پنهان‌سازی داده‌ها در فایل‌های اجرایی. این نرم‌افزار به وسیله‌ی El-Khalil نوشته شده و از مهندسی معکوس کدهای باینری برای یافتن بهترین محل برای پنهان کردن داده‌ها در فایل‌های اجرایی استفاده می‌کند. برای انجام این کار، از روش Mammon's libdisasm، که یک مجموعه‌ی کتابخانه‌ای x86 است، استفاده می‌شود. فایل‌های باینری فضای بسیار کوچکی برای پنهان‌سازی داده فراهم می‌کند. در حالی که ممکن است نسبت داده‌های پنهان شده به کل داده‌ها در یک فایل تصویر JPEG، یک به ۱۷ بایت باشد، در یک فایل اجرایی، این نسبت یک به ۱۵۰ بایت است. مسلماً تغییر فایل‌های اجرایی باید چنان با احتیاط انجام گیرد که خللی در اجرایی شدن فایل حاصل ایجاد نکند.

نرم‌افزار Hydan با بسیاری از سیستم عامل‌ها چون Linux و FreeBSD کار می‌کند. در این مثال از فایل «tar» باینری برای پنهان کردن داده‌ها به شکل زیر استفاده کرده‌ایم:

```
[root@localhost hydan]# ls -al
total 2760
drwx----- 5 1000 users 4096 Jun 9 17:42.
drwxr-xr-x 3 root root 4096 Jun 9 17:42..
-rw-r--r-- 1 root root 7 Jun 9 17:36 message.txt
-rwxr-xr-x 1 root root 150252 Jun 9 16:40 tar
[root@localhost hydan]# ./hydan tar message.txt > tar.steg
Password:
Done. Embedded 16/16 bytes out of a total possible 561 bytes.
Encoding rate: 1/201
[root@localhost hydan]# ls -al
total 2760
drwx----- 5 1000 users 4096 Jun 9 17:42.
drwxr-xr-x 3 root root 4096 Jun 9 17:42..
-rw-r--r-- 1 root root 7 Jun 9 17:36 message.txt
-rwxr-xr-x 1 root root 150252 Jun 9 16:40 tar
-rwxr-xr-x 1 root root 150252 Jun 9 17:43 tar.steg
[root@localhost hydan]# ./hydan-decode tar.steg
Password:
hideme
```

```
[root@localhost hydan]#./tar.steg -xvf hydan-0.13.tar
hydan/
hydan/CVS/
hydan/CVS/Root
hydan/CVS/Repository
hydan/CVS/Entries
hydan/msg
hydan/TODO
hydan/Makefile
```

همانگونه که مشاهده می‌کنید، یک فایل tar باینری جدید ایجاد شد. با آزمایش ویژگی‌های آن در زمینه‌ی پنهان سازی داده‌ها مشاهده می‌کنیم که این tar هم دقیقاً مانند tar باینری به شکل قانونی و درست عمل می‌کند. ابزار Rakan نه تنها اثر داده‌های پنهان شده در فایل اجرایی را نشان می‌دهد، بلکه به سادگی داده‌های پنهان را نیز آشکار می‌کند؛ به‌علاوه، این ابزار می‌تواند برای پیوست کردن امضای دیجیتال به فایل جاسازی واترمارک و تغییر نرم‌افزارهای مخرب برای گریز از شناسایی شدن به‌وسیله‌ی آنتی ویروس هم مورد استفاده قرار گیرد.

## پنهان‌سازی داده‌ها در HTML

با استفاده از نرم‌افزار <sup>۱</sup> Snow، طراحی شده به وسیله‌ی Matthew kwan می‌توان داده‌های متنی در قالب کدهای Ascii را در پایان خطوط فایل HTML و با استفاده از کاراکتر TAB یا space پنهان نمود، به گونه‌ای که این داده‌ها در نمایش فایل به وسیله‌ی مرورگرها دیده نشود. این نرم‌افزار، امکان استفاده از روش رمزنگاری به وسیله‌ی ICE<sup>۲</sup> را نیز فراهم می‌کند. نرم‌افزار Snow برای سیستم عامل Dos طراحی شد و اکنون به‌صورت نرم‌افزار منبع باز در اختیار همگان است.

همانگونه که اشاره شد، داده‌ها را به وسیله‌ی افزودن حداکثر ۷ نویسه Space پشت سرهم که بین کاراکتر Tab پراکنده شده‌اند، در متن فایل پنهان می‌شوند. این کار، امکان پنهان کردن ۳ بیت به ازای هر ۸ ستون از نویسه را فراهم می‌کند.

این ابزار، از خط فرمان به شکل زیر اجرا می‌شود:

---

<sup>۱</sup> [www.darkside.com.au/snow/](http://www.darkside.com.au/snow/)

<sup>۲</sup> Information Concealment Engine

```
C:\>snow.exe -C -m "aaaaaaaaaaaaaaaa" -p "zzzzzzzz" SpyHunter.htm
```

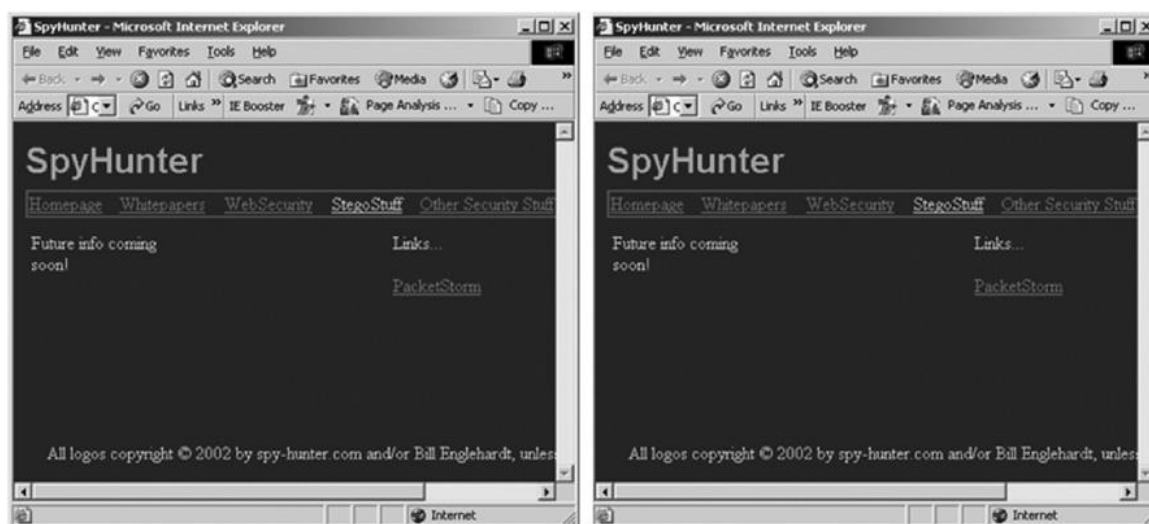
```
SpyHunterwithsnow.htm
```

```
Compressed by 50.00%
```

```
Message used approximately 4.18% of available space.
```

با مقایسه فایل اصلی HTML و فایل دارای محتوای پنهان مشاهده می‌شود که تفاوتی بین نمایش

آنها در مرورگر وجود ندارد (شکل ۳-۱۵).



شکل ۳-۱۵: HTML پیش و پس از استفاده از نرم‌افزار SNOW برای پنهان‌سازی داده‌ها

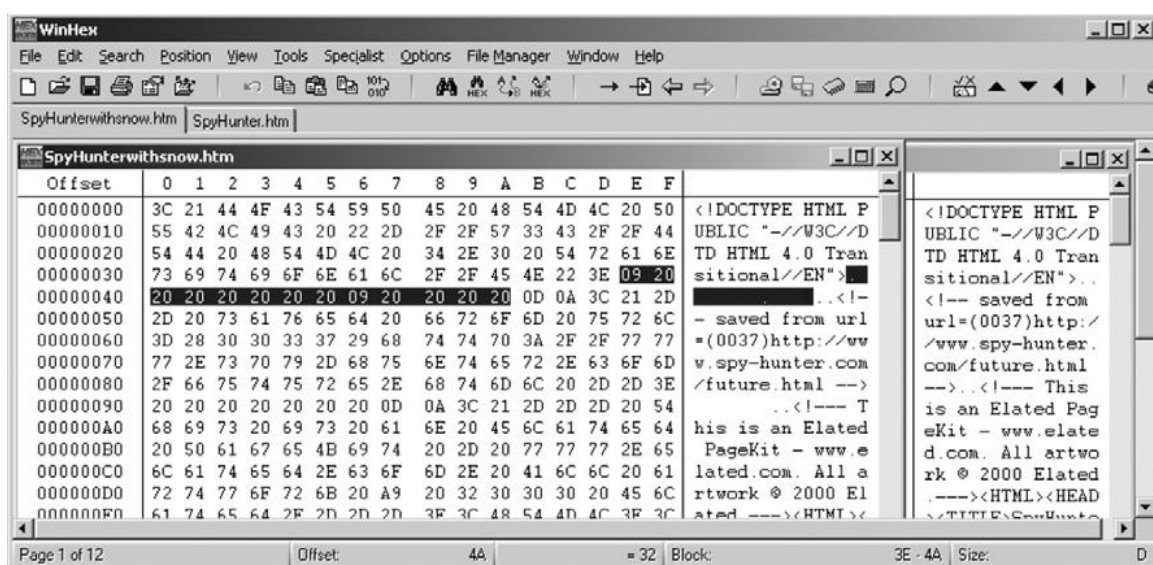
در واقع با بررسی فایل در یک ویرایشگر HTML مشاهده می‌شود که نشانه آشکاری از داده‌های

افزوده شده به شکل پنهان وجود ندارد. تنها با انجام مقایسه در سطح فایلی است که می‌توانیم نشانه‌ای از

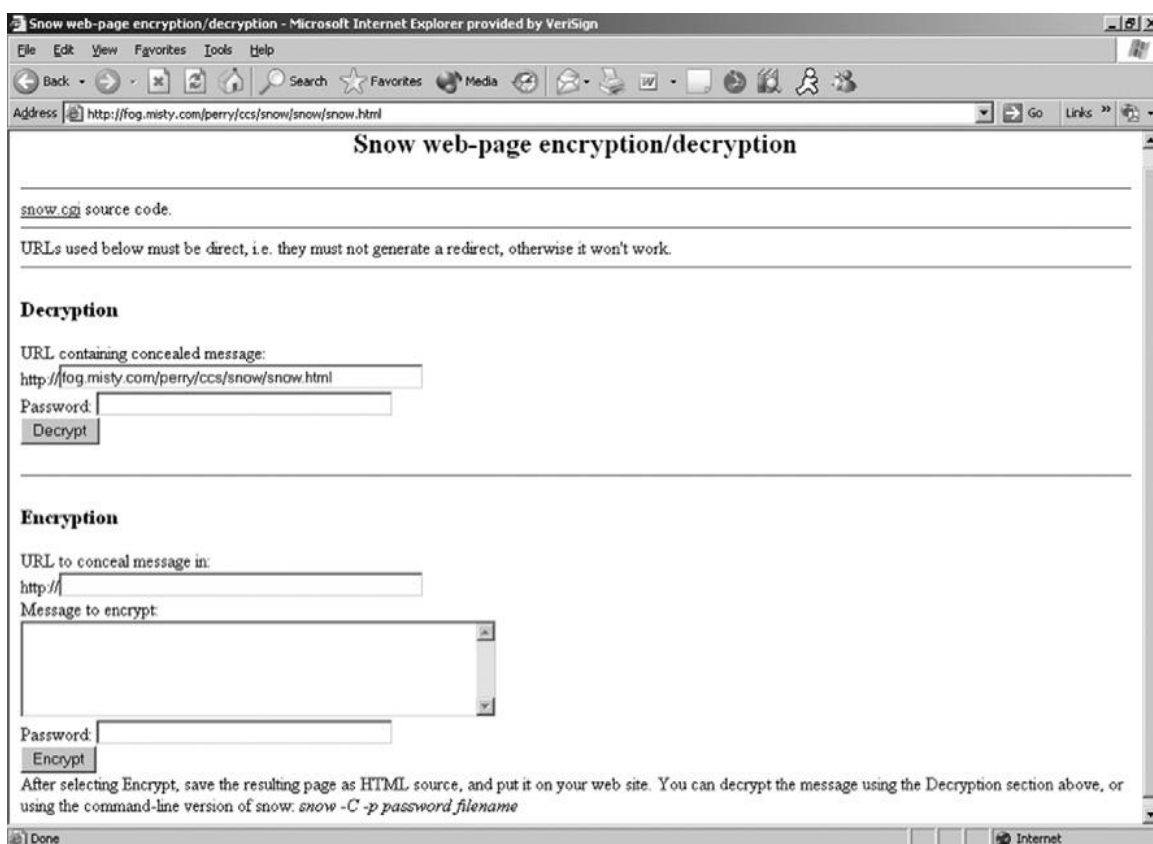
داده‌های پنهان که به شکل رشته‌ای از نویسه‌های Tab و Space در کل فایل می‌باشد را مشاهده

نماییم (شکل ۳-۱۶).





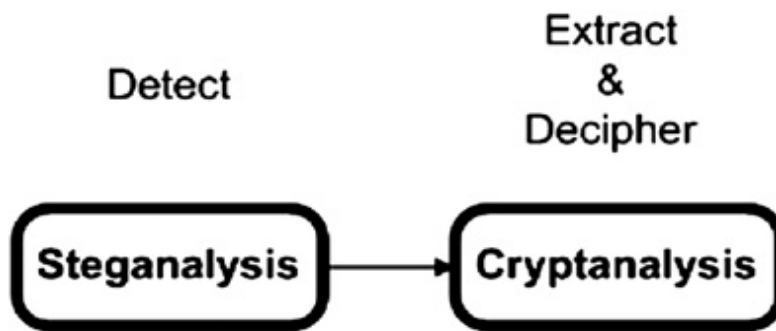
شکل ۳-۱۶: استفاده از WinHex برای مقایسه فایل‌های HTML پیش و پس از استفاده از نرم‌افزار Snow  
یک نسخه آنلاین این ابزار برای رمزنگاری و رمزگشایی فایل‌های HTML از آدرس <http://fog.misty.com/perry/ccs/snow/snow/snow.html> در دسترس است (شکل ۳-۱۷)



شکل ۳-۱۷: نسخه آنلاین نرم‌افزار Snow

## تحلیل پوشیده‌نگاری<sup>۱</sup>

تحلیل استتار دیجیتال، پروسه‌ی تشخیص مدارکی دال بر وجود داده‌های پنهان اضافه شده به وسیله‌ی روش‌های استتار یا نرم‌افزارهای مربوطه می‌باشد. اگر داده‌های پنهانی رمزنگاری شده باشند، روش‌های تحلیل رمز نیز برای رمزگشایی پیام پنهان موردنیاز است. این دستورالعمل‌ها غالباً گیج کننده است. مثلاً هنگام کار با نرم‌افزار استتار، نخست باید اقدام به تحلیل استتار، سپس تحلیل رمز نمود (شکل ۳-۱۸). هدف ایده‌آل بازرسان آشکار کردن پیام پنهان است. اما باید توجه داشت که تحلیل داده‌های پنهان رمزنگاری شده، پروسه‌ای دو مرحله‌ای است که هر یک از مراحل آن دارای تکنیک‌های متفاوتی است. تحلیل‌گر نمی‌تواند اقدام به رمزگشایی متن پنهان کند مگر این که نخست از وجود متن پنهان شده در فایل حامل مطمئن شده باشد.



شکل ۳-۱۸: پروسه تحلیل استتار

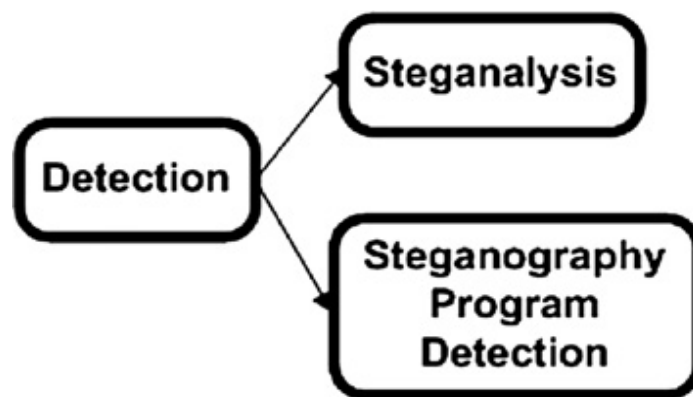
روش‌های تحلیل استتار، به شیوه‌های پنهان‌سازی به کار رفته در زمان ایجاد فایل بستگی دارد. مثلاً متن پنهان ممکن است در کل فایل حامل پراکنده شده باشد، اما نرم‌افزارهای پنهان‌سازی داده، ممکن است ردی از خود به جا بگذارند. برنامه‌نویس این گونه نرم‌افزارها ممکن است این رد را عمدی یا غیرعمدی به جای بگذارد و این کار به این خاطر انجام می‌شود که اگر گیرنده از نرم‌افزار مشابه برای تشخیص داده‌های پنهان استفاده کند، خود نرم‌افزار نخست تشخیص دهد که آیا داده پنهان شده‌ای در فایل وجود دارد یا نه. این نقص در نرم‌افزار، مزیتی برای بازرسان محسوب می‌شود. به عنوان مثال، نرم‌افزار استتار Hiderman سه کاراکتر کداسکی CDN را به آخر فایل حامل اضافه می‌کند.

معمولاً به این داده‌های ردیابی، امضا می‌گوییم. کرم‌ها و بدافزارها با عملکرد مشابه آن، معمولاً رشته‌ای از حروف که بیانگر روش به کار رفته در چگونگی کارکرد بدافزار است را به همراه دارند و این

<sup>۱</sup> Steganalysis

امضا معمولاً برای تشخیص سیستم شناسایی مهاجم<sup>۱</sup> به کار می‌رود. وقتی که بدافزار یا ویروسی منتقل می‌شود، سیستم تشخیص مهاجم می‌تواند مسئول شبکه را از این امر آگاه سازد. همین عملکرد هم در خصوص امضا در استتار به کار می‌رود. اسکنرها، پایگاه داده‌ای لیستی از نرم‌افزارهای استتار و امضای هر کدام را نگهداری می‌کنند و به وسیله‌ی آن راهی سریع و مؤثر در پویش تمام فایل‌های ماشین‌مظنون به داشتن پیام پنهان شده در فایل‌های دیگر را در اختیار کاربران می‌گذارند.

همچنین اسکنرها در کامپیوتر مظنون، به دنبال مدارکی دال بر نصب برنامه استتار می‌گردند که شامل فایل اجرایی خود نرم‌افزار، فایل‌های نصب شده به وسیله آن و یا مدخل‌های رجستری<sup>۲</sup> است. شایان توجه است که این پویش، راه دیگری برای کشف وجود نرم‌افزار استتار است نه روش دیگری برای تحلیل استتار<sup>۳</sup> (شکل ۳-۱۹). تحلیل استتار، پروسه‌ای برای شناسایی محتویات پنهان شده است نه پروسه‌ای برای شناسایی نرم‌افزار استتاری که ممکن است در کامپیوتر نصب شده باشد.



شکل ۱۹-۳: اشکال شناسایی پوشیده‌نگاری

با وجود ۲۰۰ برنامه استتار، رسیدن به این هدف بسیار مشکل به نظر می‌رسد و این بدان علت است که هر نرم‌افزار استتاری، تکنیک‌ها و روش‌های ویژه‌ی خود را برای پنهان‌سازی، رمزنگاری و پاسداری از محتوی پنهان به‌وسیله‌ی گذرواژه دارد. اگرچه تکنیک‌های پایه‌ای برای پنهان کردن محتوی وجود دارد، اما هر نرم‌افزار، آن را به روش خود در برنامه پیاده می‌کند. بسیاری از روش‌های پایه‌ای که

<sup>۱</sup> Intrusion Detection system

<sup>۲</sup> registry

<sup>۳</sup> steganalysis

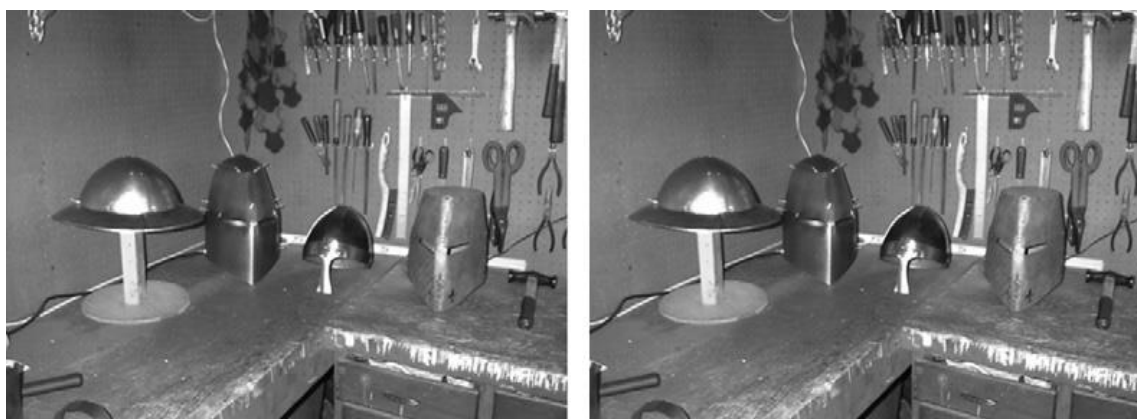
برای پنهان‌سازی داده‌ها به کار می‌روند را می‌توان به شکل وارون برای آشکارکردن محتوی پنهان و تحلیل استفاده نمود.

## تحلیل‌های غیرمعمول

تحلیل‌های غیرمعمول، شامل تکنیک‌های مورد استفاده در شناسایی تفاوت‌های بین دو فایل مشابه می‌باشد. همچنین این تحلیل شامل تکنیکی است که هنگامی که هیچ فایلی برای مقایسه وجود نداشته باشد، سایر موارد غیر معمول را در خصوص آن فایل شناسایی کند.

## ویژگی‌های فایل<sup>۱</sup>

تفاوت‌های بین ویژگی‌های دو فایل را زمانی می‌توان به سادگی تشخیص داد که هم فایل تغییر یافته و هم فایل اصلی وجود داشته باشد. از نظر دیداری، تشخیص تفاوت دو عکس موجود در شکل ۳-۲۰ غیرممکن است، اما با مقایسه مقدار checksum در نمایش لیست گونه در پوشه محتوی فایل، می‌توان به سادگی تفاوت‌ها را تشخیص داد.



شکل ۳-۲۰: فایل اصلی و فایل با داده‌های پنهان

در نمایش لیست گونه از محتویات پوشه می‌توانیم به سادگی وجود تفاوت آشکار بین دو فایل را به شکل زیر تشخیص دهیم :

D:\dir

04/04/2012 05:25p 240,759 helmetprototype.jpg

04/04/2012 05:26p 235,750 helmetprototype.jpg

مشاهده می‌شود که اندازه‌ی فایل‌ها و تاریخ ایجاد آن‌ها متفاوت است.

یک Checksum ساده هم می‌تواند امکان تشخیص تفاوت بین محتویات دو فایل را به ما بدهد:

```
C:\GNUTools>cksum a:\before\helmetprototype.jpg 3241690497 240759 a:\before\helmetprototype.jpg
C:\GNUTools>cksum a:\after\helmetprototype.jpg 3749290633 235750 a:\after\helmetprototype.jpg
```

شایان توجه است، که برخی از نرم‌افزارهای استتار، بدون این که اندازه و زمان فایل را تغییر دهد، امکان افزودن داده‌های پنهان به فایل را فراهم می‌کنند. با این وجود، با بررسی مقدار checksum دو فایل، به سرعت می‌توان به وجود اختلاف بین آن‌ها پی برد.

## ابزارهای تحلیل استتار

ابزارهای رایگان و تجاری بسیاری وجود دارد که به تحلیل‌گر اجازه‌ی تحلیل استتار را می‌دهد. بسیاری از ابزارهای رایگان، بخش محدودی از نرم‌افزارهای استتار را تشخیص می‌دهد، اما ابزارهای تجاری، کامل و فراگیرتر بوده و بسیاری از نرم‌افزارهای استتار را تشخیص می‌دهند.

بدون در نظر گرفتن کمال و فراگیری موجود در ابزارهای تجاری، نقطه مشترک تمامی این نرم‌افزارها، امکان تشخیص ویژگی در خصوص فایل‌ی است که حاوی داده‌های پنهان است. اصل فایل مظنون برای تحلیل دقیق‌تر نگه داشته می‌شود و یک کپی از آن به صورت خودکار یا دستی ایجاد می‌شود. پیشرفته‌ترین ابزارها هم امکان تحلیل دستی را برای مشاهده‌ی داده‌ها و تشخیص ناهنجاری در اختیار کاربر قرار می‌دهند.

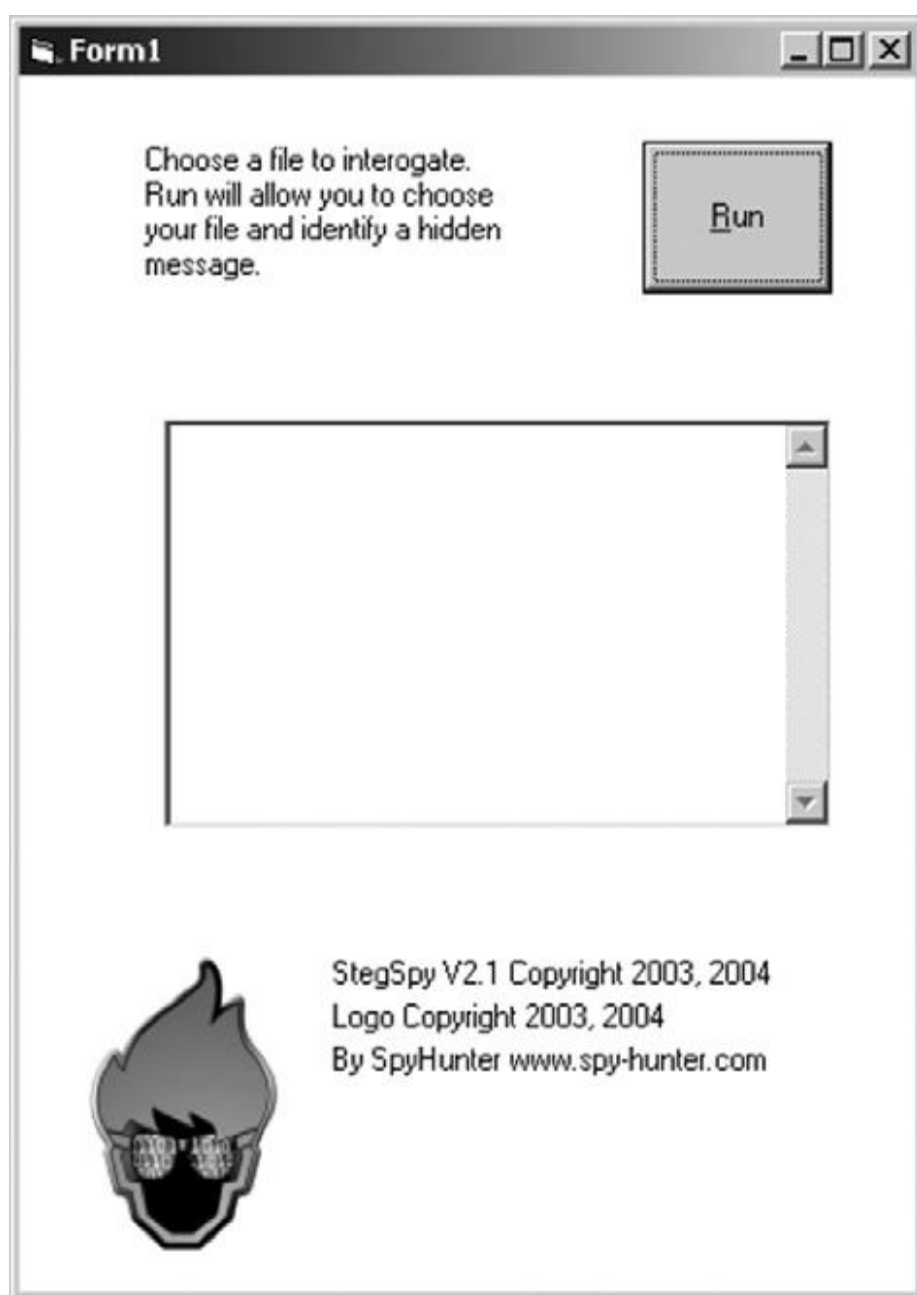
برنامه‌های استتار از روش‌های گوناگونی برای پنهان کردن داده‌ها استفاده می‌کنند. علاوه بر این، اجرای این روش‌ها در نسخه‌های مختلف نرم‌افزارهای استتار می‌تواند متفاوت باشد. این تفاوت‌ها باعث پیچیده شدن پروسه تحلیل استتار می‌گردد. در برخی از نرم‌افزارها برای افزایش پیچیدگی، حتی از روش‌های مختلف رمزنگاری در زمان پنهان‌سازی داده‌ها استفاده می‌شود و این روش‌ها می‌تواند در مورد چندین نوع فایل هم به کار روند. تمام این موارد باید در تحلیل فایل، قابل تشخیص بوده تا تحلیل‌گر بتواند به دقت نرم‌افزار و نسخه درست آن را تشخیص دهد. با دانستن این موارد، تحلیل‌گر با به کارگیری مهندسی معکوس، روش به کار رفته در پنهان‌سازی داده و در نهایت پیام پنهان را آشکار می‌کند.

## نرم‌افزار Steg spy

نرم‌افزار stag spy، نرم‌افزار رایگان تحلیل امضاء دیجیتال برای تشخیص مدارک دال بر وجود داده‌های پنهانی است. این نرم‌افزار رایگان را می‌توانید از آدرس <http://www.spy-hunter.com> دانلود نمایید. این نرم‌افزار به زبان ویژوال بیسیک نوشته شده و به وسیله‌ی بیشتر نسخه‌های سیستم عامل ویندوز پشتیبانی می‌شود. نصب آن ساده و تنها شامل یک فایل اجرایی است؛ پس به سادگی این فایل را کپی و اجرا نمایید. صفحه شروع آن مثل شکل ۳-۲۰ است.

### به‌کارگیری Stag spy

به محض اجرا، شروع به تحلیل فایل‌های مظنون به داشتن محتویات پنهان براساس لیستی از امضاهای موجود در خود نرم‌افزار می‌نماید. چنانچه محتویات پنهان شده‌ای کشف شد، فایل حاوی این داده‌ها، نرم‌افزار مورد استفاده در پنهان‌سازی آنها و محل خود داده‌های پنهان شده در فایل را به کاربر گزارش می‌دهد. شکل ۳-۲۱ گزارش حاصل از کشف داده‌ها پنهان و نرم‌افزار پنهان‌سازی داده Masker را نشان می‌دهد.



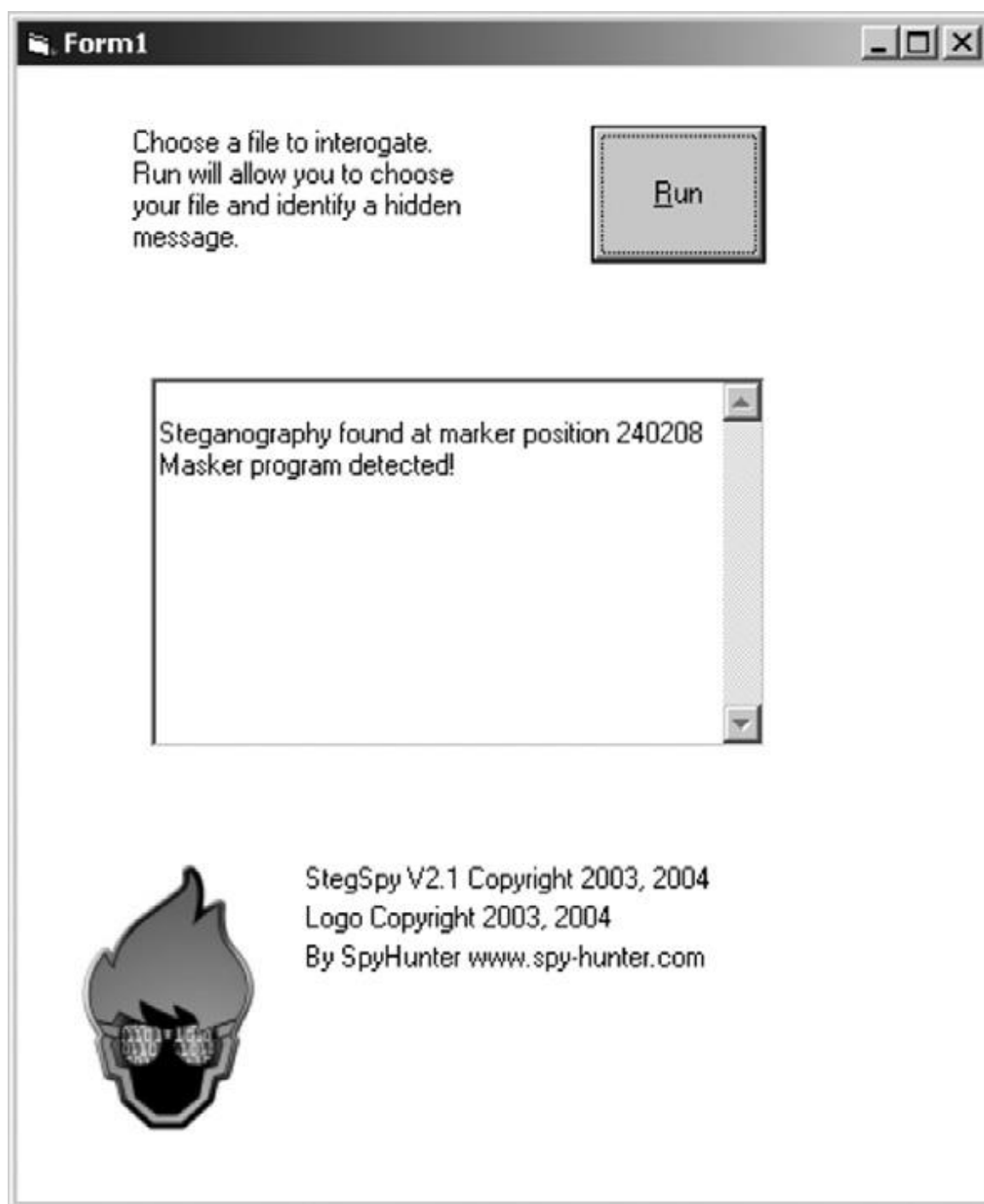
شکل ۳-۲۱: پنجره‌ی شروع نرم‌افزار StegSpy

پس از پیدا کردن فایل مظنون، می‌توان آن را در نرم‌افزارهایی چون Winhex برای بررسی بیشتر محتویات باز کرد وجود برخی نشانگرها، بیانگر ردپای نرم‌افزار نهان‌سازی است که پیشتر به وسیله‌ی steg spy گزارش شده بود و می‌توان آن را برای کشف و آشکارسازی محتوی پنهان شده بررسی کرده‌ها در شکل ۳-۲۱ آشکارسازی محتویات پنهان به وسیله‌ی این نرم‌افزار گزارش شده است. در این مثال

~~~~~

steg spy وجود داده‌های پنهان را در موقعیت ۲۴۰۲۰۸ نشان می‌دهد. چنانچه فایل را در ویرایشگر Winhex باز کنیم سه ستون را همانند شکل ۳-۲۲ مشاهده می‌کنیم. ستون سمت چپ بیانگر فایل بوده و هر یک به نشانگر ویژه‌ای اشاره می‌کند. با اسکرول به سمت پایان فایل مقدار آفست‌ها هم افزایش می‌یابند. پنجره‌ی وسط نرم‌افزار، داده‌ها را در مبنای ۱۶ نشان می‌دهد. بخش سمت راست، هم داده‌ها را در قالب کد Ascii نشان می‌دهد، همانند همان کاری که نرم‌افزار Notepad برای نمایش فایل متنی در ویندوز انجام می‌دهد.





شکل ۲۲-۳ گزارش نرم‌افزار **steg spy** پس از تشخیص داده‌های پنهان به وسیله‌ی نرم‌افزار **Masker**

شکل ۲۲-۳ محل نهان‌سازی داده‌ها در فایل و نرم‌افزار پنهان‌سازی داده‌ها که به وسیله‌ی **steg spy** کشف شده را نشان می‌دهد و برای بازرسانی که در جستجوی یافتن و آشکار کردن داده‌های پنهان در

فایل‌ها هستند بسیار مفید است. اما مشخص کردن محل داده‌های نهان تنها گام نخست کارشان است. اگر داده‌های نهان رمزنگاری هم شده باشند، بازرسان می‌بایست یا گذرواژه را کشف کنند تا بتوانند داده‌های پنهان را آشکار کنند یا مهندسی معکوس را بر روی داده‌های کشف شده انجام دهند. تا این مرحله اطلاعات مفیدی را گردآوری کردیم. هم به وجود داده‌های پنهان پی بردیم و هم مکان داده‌های پنهان در فایل را مشخص کردیم و هم نرم‌افزار مورد استفاده را پیدا کردیم. دست‌کم با دانستن نوع نرم‌افزار می‌توان به روش به کار رفته در پنهان‌سازی داده‌ها پی برد. پس با به کارگیری این روش و دانستن تکنیک به کار رفته در پنهان‌سازی داده‌ها، اقدام به آشکار سازی محتوای پنهان می‌نمایم.

### نرم‌افزار Stegdetect

این نرم‌افزار منبع باز به وسیله‌ی نیلز پروس<sup>۱</sup> یکی از پیشگامان پرآوازه‌ی تحقیقات تحلیل انتشار نوشته شد. او تحقیقات مستقلی را در خصوص حملات ۱۱ سپتامبر به پایان رساند و نتایج آن را هم در مقالات و هم در وب سایت خود در دسترس همگان قرار داده است. با کمال تعجب قوانین ایالتی میشیگان به وی اجازه‌ی ادامه‌ی تحقیقات را نداد و بنابراین به مرکز بسیاری از مباحثه‌ها تبدیل شد. در نتیجه مجبور شد تا ادامه‌ی فعالیت سایتش را به کشور هلند منتقل کند. آدرس سایتش <http://niels.xtdnet.nl/stego> است.

تحقیقات نیلز برپایه‌ی تحلیل‌های آماری بوده، پس عجیب نیست که نرم‌افزار Stegdetect هم از اول برای تحلیل فایل‌های Jpeg طراحی شده باشد؛ بنابراین می‌توان محتویات پنهان شده به وسیله‌ی نرم‌افزارهای JSteg, JPHide, OutGuess, Invisible Secrets, F5, appendX, Camouflage را تشخیص دهد.

فرمت‌های jpeg و mpeg از تابع Dct برای فشرده‌سازی محتویات استفاده می‌کنند و این فشرده‌سازی بر کاهش تعداد بیت‌های موردنیاز برای نمایش تصویر متمرکز می‌شود و این کاهش را با مشخص کردن بلوک‌های ۸\*۸ پیکسلی یکسان نزدیک به هم و حذف افزونگی آنها با استفاده از تقریب-های ریاضیاتی به دست می‌آورد (در مورد فایل‌های تصویری با فرمت mpeg فشرده‌سازی با حذف فریم-های یکسان و پشت سرهم صورت می‌گیرد). این روش فشرده‌سازی را روش با اتلاف می‌نامند؛ زیرا در پروسه‌ی فشرده‌سازی، برخی از داده‌ها از بین می‌روند ولی موجب کاهش آشکاری در کیفیت تصویر یا فیلم نمی‌شود. نرم‌افزار stegspy این‌گونه طراحی شده که نخست فرکانس ضرایب جدول DCT فایل را

<sup>۱</sup> Neils Provos

به دست آورده سپس این ضرایب را با فرکانس موجود در داده‌های فایل مضمون به داشتن داده‌های پنهان مقایسه می‌کند. آشکار است که استفاده از این روش مستلزم مدل کردن داده‌ها و دانش قبلی می‌باشد. بنابراین بیشتر تحلیل‌های آقای نیلز مبتنی بر الگوریتم آدرس است. این‌گونه تحلیل‌ها را تحلیل X می‌نامند. وجود انحراف معیار زیاد بالاتر از حد میانگین در تحلیل بیانگر احتمال وجود داده‌هایی در فایل می‌باشد که به شیوه‌های استتار در فایل درج شده‌اند.

برای نصب، نرم‌افزار را از آدرس <https://www.backtrack-linux.com/forensics-auditor/> دانلود نمایید. فایل Zip شامل نسخه گرافیکی و نسخه دستوری نرم‌افزار می‌باشد.

### به کارگیری نرم‌افزار steg detect

نرم‌افزار stegdetect با آنالیز تصویر به دنبال محتوی پنهان در فایل عکس می‌گردد. این نرم‌افزار تحلیل آماری را برای بررسی وجود داده‌های پنهان در فایل را انجام داده و تلاش می‌کند تا برنامه‌ای که برای پنهان‌سازی داده‌ها به کار رفته را مشخص نماید. اگر آن را از خط فرمان اجرا نمایید سوئیچ‌های زیر را در اختیار دارید:

- q. تنها فایل‌های عکس که به نظر می‌رسد داده‌های پنهانی دارند را گزارش می‌دهد.
- r. برای کاهش احتمال خطا در تشخیص فایل‌های سالم به عنوان فایل با محتویات پنهان هدر را بررسی می‌کند؛ اگر این گزینه فعال شود فایل‌های عکس که فیلد توضیح دارند به عنوان فایل با داده‌های پنهان تشخیص داده نمی‌شود.
- s. حساسیت الگوریتم شناسایی داده‌های پنهان را تعیین می‌کند. هر چه عدد بزرگتر باشد حساسیت الگوریتم هم بیشتر می‌شود. مقدار پیش فرض برابر یک است.

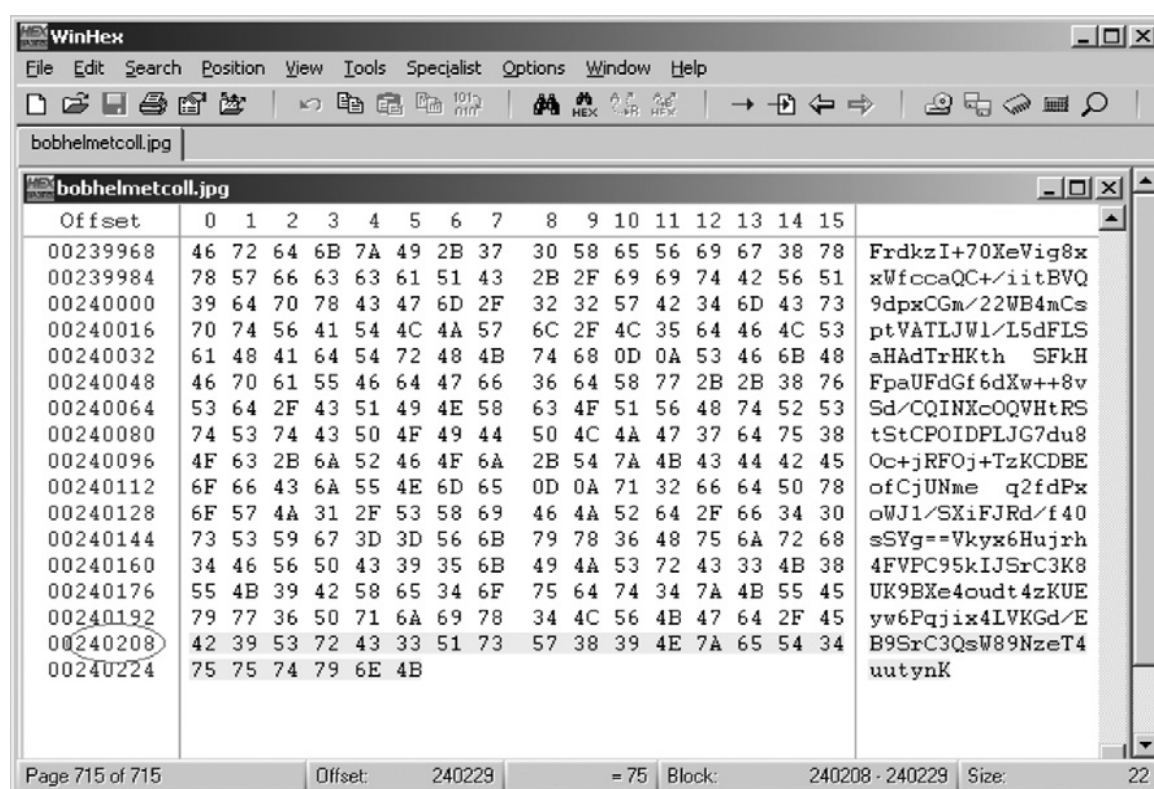
اگر پس از اجرای نرم‌افزار، موردی کشف شد میزان اطمینان از وجود داده‌های پنهان در فایل با نمایش چند ستاره مشخص می‌شود. سه ستاره بیانگر اطمینان زیاد از وجود داده‌های پنهان در فایل است. تحلیل‌های بیشتر نشان می‌دهد که این نرم‌افزار در بررسی فایل‌های عکس دیجیتال با کیفیت بالا مثل عکس‌هایی که با دوربین‌های دیجیتال گرفته موفق‌تر عمل می‌کند. در مثال زیر کاربرد آن را برای پوشش تمام فایل‌های jpeg در پوشه جاری و بررسی وجود داده‌های پنهان در آنها و در صورت وجود داده‌ی پنهان، نرم‌افزار پنهان‌سازی که داده‌ها را پنهان نموده را بررسی می‌کنیم. حساسیت الگوریتم تشخیص نرم‌افزار را باز، یک به ده افزایش می‌دهیم و نتایج حاصل به شکل زیر است:

```

D:\>stegdetect -tjopi -s10.0 *.jpg
bobhelmetcollwithhidden.jpg: jphide(**)
Corrupt JPEG data: 30 extraneous bytes before marker 0xdb
bobhelmetprototype.jpg: error: Quantization table 0x00 was not
defined
Corrupt JPEG data: 30 extraneous bytes before marker 0xdb
bobhelmetprototype_withdifferentfileanddifferentpassword.jpg: error:
Quantization table 0x00 was not defined
bobhelmetprototypewithanotherhidden.jpg: jphide(**)
Corrupt JPEG data: 26 extraneous bytes before marker 0xd9
bobhelmetprototypewithhidden.jpg: jphide(**)
familyonthecouchnormalpost.jpg: jphide(***)
securitdaemonlogowithhiddenfile.jpg: skipped (false positive likely)
securitydaemonlogo.jpg: invisible[4](***) skipped (false positive
likely)

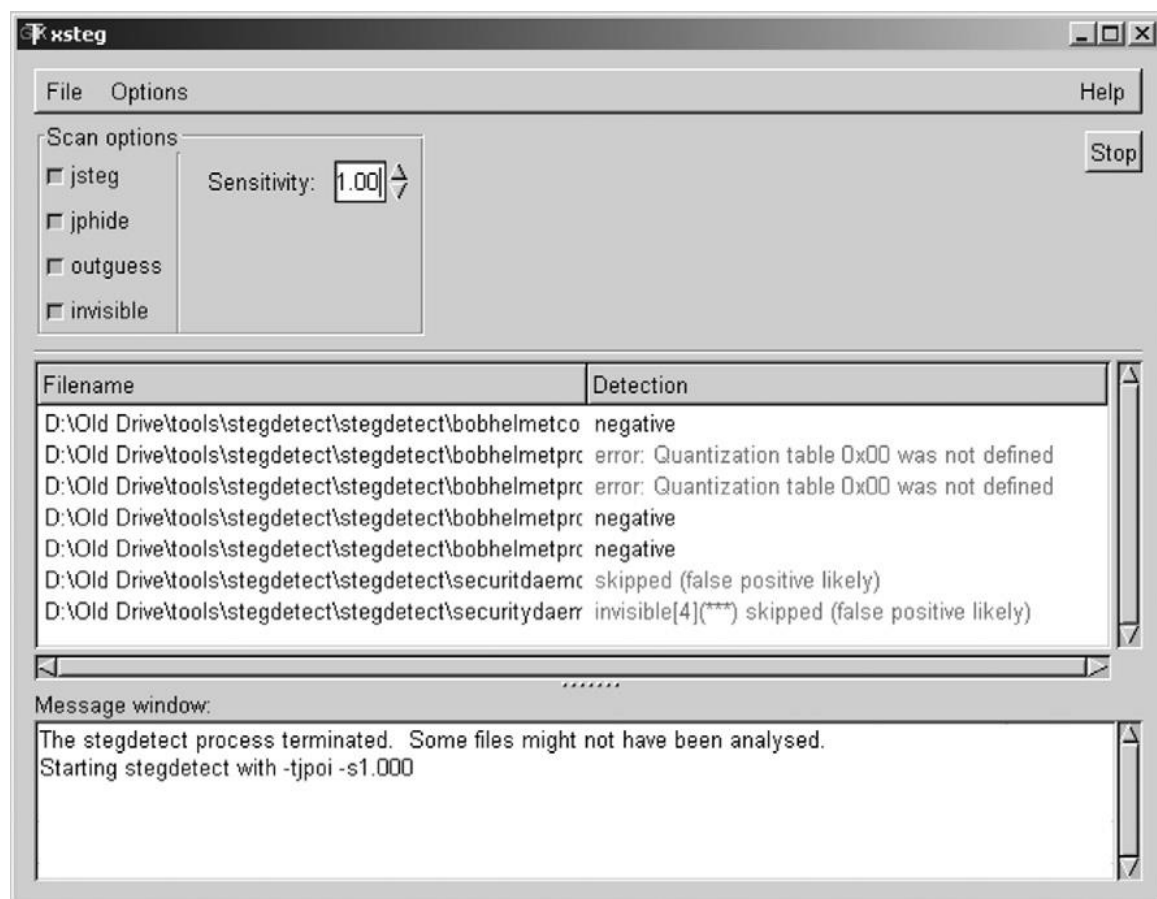
```

همانگونه که پیشتر گفتیم، میزان اطمینان این نرم‌افزار از وجود داده‌های پنهان با مقدار ستاره‌هایی در کنار نام نرم‌افزار پنهان‌سازی نشان داده می‌شود. در مثال پیشین **stegdetect** چند فایل با احتمال حضور داده‌ی پنهان و نرم‌افزار پنهان‌سازی را حدس زده است. همین پویش را با مقدار پیش فرض نرم-افزار انجام می‌دهیم، در نتیجه حساسیت الگوریتم همان عدد یک پیش فرض است. اگر مقدار حساسیت را بیش از حد زیاد کنید تعداد زیادی فایل به اشتباه حاوی داده‌های پنهان اعلام شده و گزارش نرم‌افزار از واقعیت فاصله زیادی می‌گیرد (عکس ۲- ۲۴).

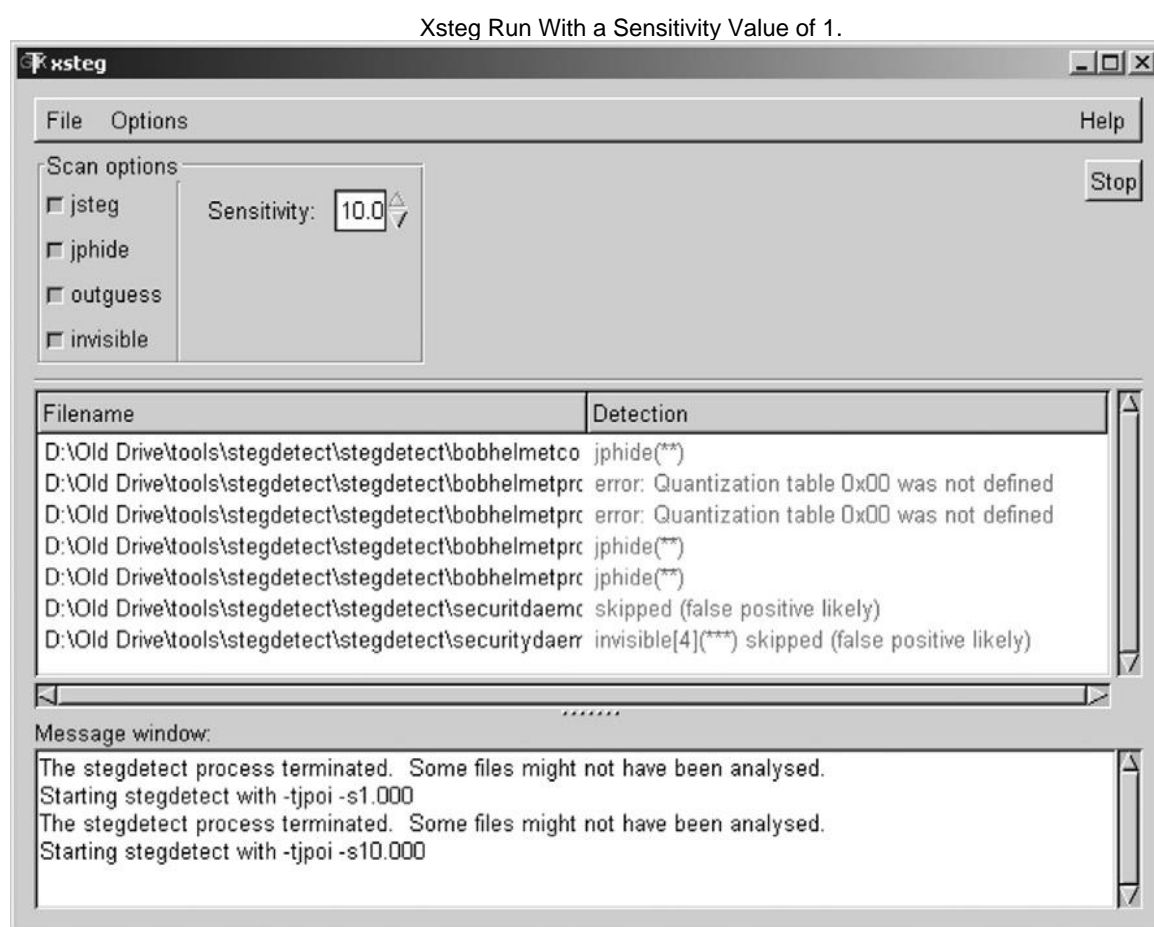


در شکل ۳-۲۳ فایل مضمون به داشتن داده‌های پنهان را در Winhex مشاهده می‌کنید.

شکل ۳-۲۴ اجرای این نرم‌افزاری با میزان حساسیت پیش فرض برابر یک در هر دو مثال قبل، پوشه جاری پس از فایل‌های jpeg با داده‌های پنهان بود که به وسیله‌ی نرم‌افزارهای گوناگون پوشیده‌نگاری تولید شده بود. stegdetect تعداد قابل توجهی از آنها را تشخیص نداده و در گزارش پایانی خود اعلام نکرد. به علت استفاده از تکنیک‌های پیچیده‌ی فشرده‌سازی در فایل‌های jpeg، این‌گونه فایل‌ها برای تحلیل و بررسی بسیار مشکل هستند؛ به علاوه برخی برنامه‌های پوشیده‌نگاری هیچ ردپایی از خود به جای نمی‌گذارند و نشانگر ویژه‌ای دال بر استفاده از نرم‌افزار هم به فایل حاوی داده‌های پوشیده اضافه نمی‌کنند؛ بنابراین در این وضعیت، نرم‌افزار stegdetect تنها و موثرترین ابزار تحلیل فایل jpeg بدون در اختیار داشتن فایل اصلی برای مقایسه است (عکس ۲-۲۴).



شکل ۳-۲۴ اجرای نرم‌افزار stegdetect با درجه حساسیت برابر یک.



شکل ۳-۲۵ اجرای نرم‌افزار با بالاترین درجه حساسیت الگوریتم برابر ده است.

## چیکده

در این فصل تاریخچه‌ی استتار دیجیتال را از اواسط دهه ۱۹۹۰ تا امروز مرور کردیم. نرم‌افزارها و روش‌های رایج پنهان‌سازی داده‌ها، نشان‌دهنده‌ی طیف گسترده‌ای از روش‌های استتار است. این روش‌ها شامل پنهان کردن داده‌ها در عکس، فایل‌های HTML و فایل‌های اجرایی است. در فصل‌های آینده راه‌های نوین پنهان‌سازی داده در سیستم‌های عامل، فایل‌های چندرسانه‌ای و ابزارهای همراه را بررسی می‌کنیم.

~~~~~



## فصل چهارم

### پنهان سازی داده ها در فایل های چندرسانه ای

#### مروری بر چندرسانه ای

موسیقی دیجیتال، پخش برنامه های رادیویی به وسیله اینترنت، پخش زنده یا ضبط شده هم اندیشی ها در اینترنت، مکالمات تصویری و ارسال فایل های ویدئویی، راه های ارتباطی زندگی ما را دگرگون کرده اند و به بخش جدایی ناپذیر از ماهیت سازمان ها تبدیل شده اند. ما از این روش ها برای انتقال طرح ها و ایده ها، آموزش کارمندان، ارتباط با مشتریان و صد البته برای سرگرمی استفاده می کنیم. پرسش اینجاست که آیا فایل های چندرسانه ای دیجیتال برای ما خطری به همراه دارد؟ آیا از این کانال ها می توان به عنوان پوشش برای انتقال پنهانی اطلاعات استفاده کرد؟ آیا امکان فیلتر نمودن سرمایه های فکری و پیشگیری از نشت آن به بیرون سازمان، همزمان با اشتراک اطلاعات داخلی در سازمان وجود دارد؟ آیا می توان از این فایل ها برای انتقال دستورات و اطلاعات کنترلی یا دروازه های برای فعال سازی فناوری های مورد نیاز مهاجمان به روش پیشرفته استفاده کرد؟ از این گذشته، از آنجا که اندازه ی فایل های چندرسانه ای بسیار بیشتر از فایل عکس است، آیا این بدان معناست که این فایل ها محل بزرگ تری برای پنهان سازی داده ها - هایی است که می توان با استفاده از کاستی های موجود در ساختار فایل چندرسانه ای به عنوان فایل حامل منتقل نمود؟ آیا سیستم شنیداری انسان<sup>۱</sup> آن قدر حساس است که تغییرات ایجاد شده بر اثر جاسازی داده های پنهان در این گونه فایل ها را تشخیص دهد؟

در این فصل، اشکال ساده و بدوی پنهان سازی در فایل های چندرسانه ای دیجیتال را بررسی، سپس به روش های نوین پنهان سازی داده ها برای پاسخ به این پرسش ها می پردازیم.

---

<sup>۱</sup> Human auditory system

## پنهان‌سازی داده‌ها در صوت دیجیتال

تعداد قابل توجهی از تحقیقات نشان می‌دهند که عکس‌های دیجیتال می‌توانند فایل‌های حامل اطلاعات پنهان شده باشند. اما سیستم شنیداری انسان با داشتن ویژگی حس شنیداری دقیق و حساس، امکان پنهان‌سازی داده‌ها را مشکل می‌نماید. اما بنا به نظر Bender, Gruhl و Morimoto در 1996 (میلادی)، با این که سیستم شنیداری انسان طیف گسترده‌ای دارد اما دامنه‌ی تشخیصی محدودی دارد؛ در نتیجه صدای بلند، صداهای آرام‌تر را می‌پوشاند. از این گذشته، سیستم شنیداری انسان قادر به درک حالت مطلق نیست و تنها می‌تواند حالت نسبی را تشخیص دهد. این محدودیت‌ها پایه و اساس روش‌های پنهان‌سازی به‌کاررفته برای فریب حس شنیداری انسان را تشکیل می‌دهد. در ضمن چند مزیت دیگر پنهان‌سازی داده‌ها در فایل‌های صوتی به شرح زیر است:

(۱) اندازه‌ی این فایل در قامت فایل حامل، نسبت به قبل بزرگ‌تر شده‌اند و گنجایش مقادیر زیادی داده‌ی اضافی را دارند (به عنوان مثال، توانستیم کل کارهای شکسپیر را تنها در یک آواز ۸ دقیقه‌ای پنهان کنیم).

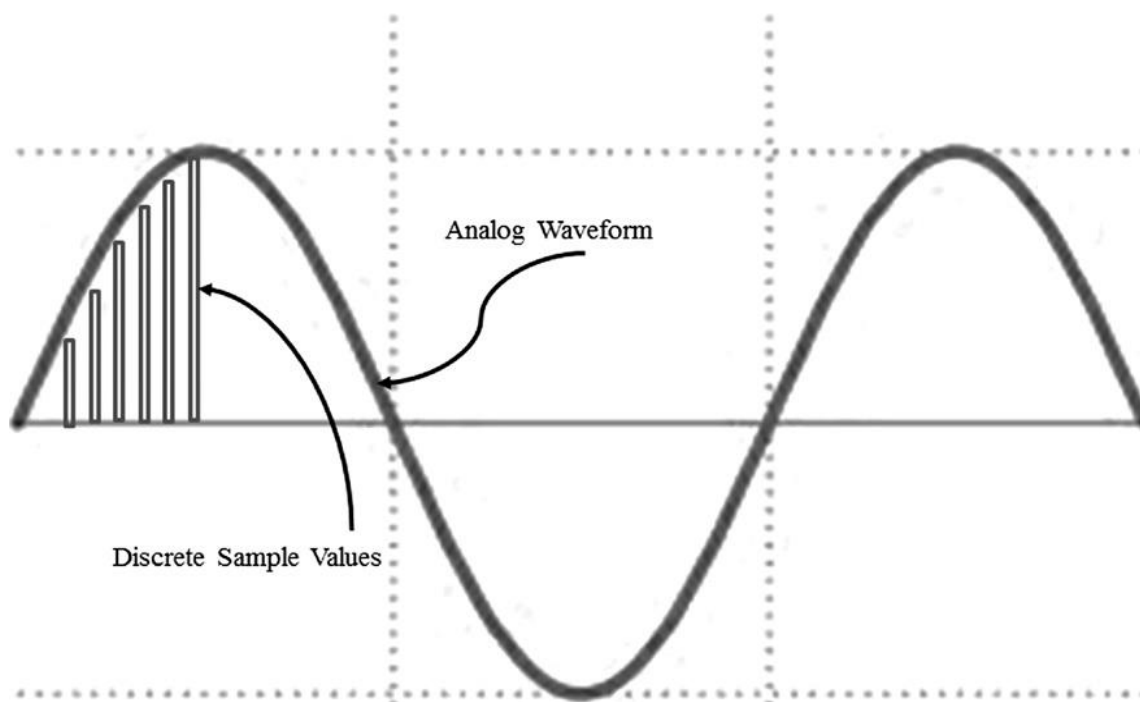
(۲) انتشار و استفاده روزانه از فایل‌های صوتی mp3 با پیدایش iPod و سایر ابزارهای پخش موسیقی، موجب تولید انبوهی از فایل‌های صوتی شده است که در سراسر جهان دست به دست می‌شوند. در گذشته پیدا کردن فایل‌هایی که حاوی داده‌های پنهان بود به پیدا کردن سوزن در انبار کاه شبیه بود. شاید پیدا کردن سوزن در انبار کاه به مراتب آسان‌تر بود؛ ولی در حال حاضر تشبیه بهتر، پیدا کردن کاه در انبار کاه است.

برخی از کارهای اولیه در زمینه‌ی پنهان کردن داده‌ها در فایل‌های صوتی، بیشتر بر بی‌تاثیر بودن بر سیستم شنیداری انسان به خاطر جاسازی داده‌های اضافه متمرکز بودند و تلاشی برای عدم تشخیص و شناسایی خود داده‌های پنهان در فایل از خود نشان نمی‌دادند. این روش‌های پنهان‌سازی برای گول زدن حس شنوایی انسان به خوبی عمل می‌کردند. با این حال، وقتی که به این اشکال ساده از پنهان‌سازی با روش‌های آماری نگاهی بیندازیم، به توانایی خود در شناسایی ساده‌ی آن‌ها پی می‌بریم. اما پرسش اصلی این است که آیا کسی به دنبال این‌گونه داده‌ها می‌گردد؟

## جاسازی داده‌ها در فایل صوتی به شکل ساده (رویکرد غیر قابل مشاهده)

یکی از روش‌های ابتدایی پنهان‌سازی داده‌ها، فایل‌های صوتی خام مانند فایل‌های با قالب wav را هدف قرار داده بود، زیرا معمول‌ترین روش دیجیتالی کردن صوت آنالوگ، بر پایه‌ی کار دکتر نایکویست است. او

وقتی که در اوایل دهه ۱۹۲۰ میلادی در آزمایشگاه Bell کار می‌کرد، به این نتیجه رسید که لازم نیست تمام شکل موج آنالوگ را برای انتقال به صورت کامل در نظر بگیریم بلکه نمونه‌برداری از سیگنال آنالوگ، اجازه تولید دوباره‌ی شکل موجی از سیگنال نزدیک به شکل موج صوت اصلی را می‌دهد (شکل ۴-۱).



شکل ۴-۱: موج آنالوگ با نمونه‌های گسسته

نایکویست همچنین به این نتیجه رسید، که برای تولید دوباره‌ی صوت اصلی و با کیفیت، می‌بایست نرخ نمونه‌برداری دو برابر پهنای باند موج اصلی باشد. بعدها، این کشف، پایه و اساس استاندارد مدولاسیون کد پالس<sup>۱</sup> در تبدیل صدای آنالوگ به داده دیجیتال شد. این بدان معناست که سیگنال صوتی 4 KHz باید ۸ هزار بار در ثانیه نمونه‌برداری شود.

برای تولید صدای موسیقی ضبط‌شده با کیفیت بالا، به گونه‌ای که تمام طیف‌های صوتی قابل شنیدن توسط انسان (5/22 Khz) را پوشش دهد، می‌بایست سیگنال ۴۴۱۰۰ بار در ثانیه نمونه‌برداری شود. به همین دلیل، موسیقی معمولاً به صورت استریو ضبط می‌شود (دو کانال صوتی و هر کانال ۴۴۱۰۰ بار در ثانیه نمونه‌برداری می‌شود). این بدان معناست که به ازای هر ثانیه از موسیقی استریو، ۸۸۲۰۰ نمونه‌ی دیجیتالی ضبط می‌شود. پس از ضبط، می‌توان تقریباً سیگنال آنالوگ اصلی ضبط‌شده را

<sup>۱</sup> Pulse code modulation

به وسیله‌ی تبدیل دیجیتال به آنالوگ دوباره تولید کرد. با این وجود، ممکن است داده‌های نمونه‌برداری شده، ابزار فعالیت‌های پنهان‌سازی داده‌ها نیز قرار گیرند و این مسئله مجموعه‌ای غنی از امکانات را در پیش رو ما قرار می‌دهد. دانستن این که هر کدام از ۸۸۲۰۰ نمونه در ثانیه، معمولاً به صورت عدد علامت‌دار ۱۶ بیتی و بین دو عدد ۳۲۷۶۷+ و ۳۲۷۶۸- ذخیره می‌شود، مسئله را جالب‌تر می‌کند. بنابراین، تغییر کم ارزش‌ترین بیت، تنها تغییر بسیار ناچیزی در ضبط آنالوگ ایجاد خواهد کرد و این نکته، روش استتار به روش تغییر کم‌ارزش‌ترین بیت برای پنهان کردن داده‌ها را به کاندیدای ایده‌آل تبدیل کرده است.

فایل صوتی Wav قالب خاص و مشخصی دارد که بسیار انعطاف‌پذیر است و هنوز هم مطابق استانداردهای امروزی ضبط و پخش دیجیتالی است. بیشتر این فایل‌ها قالب تقریباً ساده‌ای دارند که تنها دو بخش دارد: بخش داده‌های مربوط به فرمت و بخش داده‌های صوت ضبط‌شده. در شکل ۴-۲ قالب یک فایل را برای نمونه آورده‌ایم. در این مثال طول هر فیلد را مشخص و توضیح کوتاهی درباره‌ی هر کدام می‌دهیم؛ سپس با دقت به چگونگی نمایش اعداد در می‌یابیم که هر عدد را چگونه باید تفسیر کنیم.

| Value |    |    |    | Description                | Endian | Value |    |    |    | Description               | Endian |
|-------|----|----|----|----------------------------|--------|-------|----|----|----|---------------------------|--------|
| R     | I  | F  | F  | File ID                    | Big    | 02    | 00 |    |    | Channels = 2<br>or Stereo | Little |
| 28    | 5c | 67 | 00 | Segment Size               | Little | 44    | ac | 00 | 00 | 44,100 Samples / Sec      | Little |
| W     | A  | V  | E  | Audio Type                 | Big    | 10    | b1 | 02 | 00 | Byte Rate                 | Little |
| f     | m  | t  |    | Start Format Segment       | Big    | 04    | 00 |    |    | Block Align = 4 Bytes     | Little |
| 10    | 00 | 00 | 00 | Modulation Method =<br>PCM | Little | 10    | 00 |    |    | Bits Per Sample = 16      | Little |
| 01    | 00 |    |    | Quantization = Linear      | Little | d     | a  | t  | a  | Start Data Segment        | Big    |
|       |    |    |    |                            |        | 04    | 5c | 67 | 00 | Size of Data Segment      | Little |

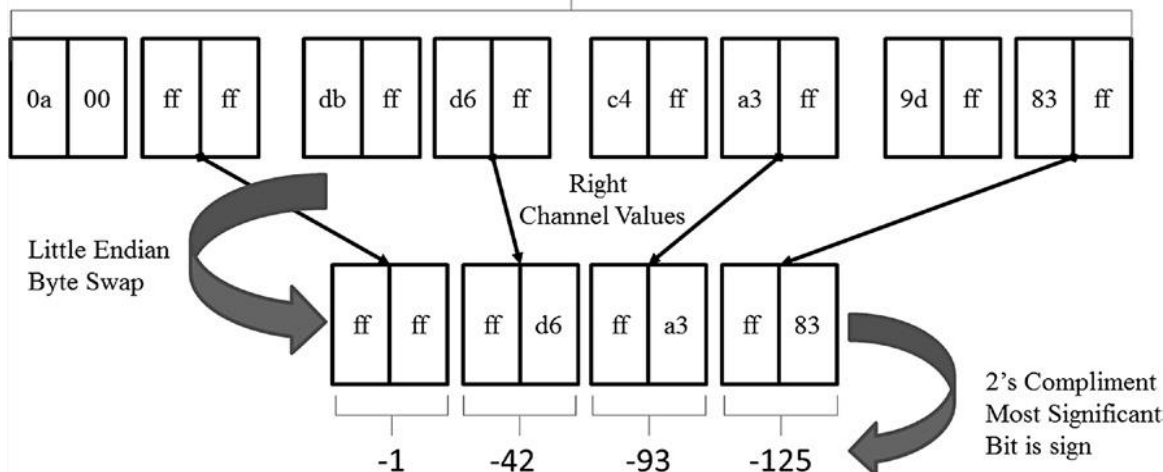
  

| Offset   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | A  | B  | C  | D  | E  | F  |                 |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------------|
| 00000000 | 52 | 49 | 46 | 46 | 28 | 5C | 67 | 00 | 57 | 41 | 56 | 45 | 66 | 6D | 74 | 20 | RIFF(\g WAVEfmt |
| 00000010 | 10 | 00 | 00 | 00 | 01 | 00 | 02 | 00 | 44 | AC | 00 | 00 | 10 | B1 | 02 | 00 | D~ ±            |
| 00000020 | 04 | 00 | 10 | 00 | 64 | 61 | 74 | 61 | 04 | 5C | 67 | 00 | 00 | 00 | 00 | 00 | data \g         |

شکل ۴-۲: هدر فایل موج نمونه

تعداد بایت‌های هر نمونه و تعداد بیت‌های استفاده‌شده.

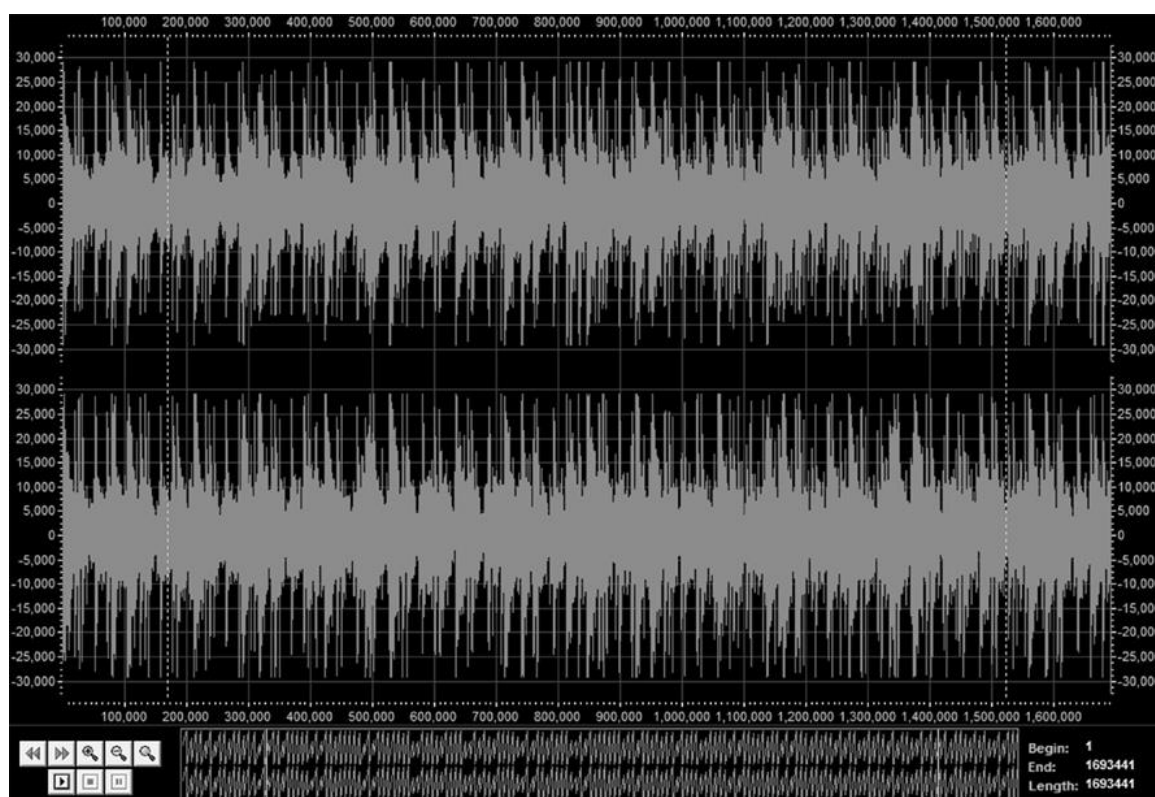
1<sup>st</sup> 16 bytes of the wave form samples (8 samples, 4 left channel and 4 right channel)



|          |                         |                         |                   |
|----------|-------------------------|-------------------------|-------------------|
| 00000030 | 0A 00 FF FF DB FF D6 FF | C4 FF A3 FF 9D FF 83 FF | ÿÿ0ÿ0ÿÄÿÿ ÿÿ      |
| 00000040 | 2F FF 8C FF D3 FE E5 FF | AB FE 58 00 ED FE 8D 00 | /ÿÿ0pây<pK ip     |
| 00000050 | 3A FF 7C 00 B8 FF 5E 00 | 22 01 50 00 6D 04 2B 00 | :ÿÿ ÿÿ " P m +    |
| 00000060 | 55 0A CB FF 08 13 44 FF | B5 1D B1 FE 88 28 02 FE | U Eÿ Dÿ ±(b       |
| 00000070 | B0 31 3A FD 57 37 91 FC | A3 3A 17 FC 86 3C AA FB | *1.ÿW7 üÿ: üÿ<âü  |
| 00000080 | 5E 3D 68 FB B4 3D 11 FB | 33 3E CC FA B5 3E B5 FA | ^=hü' = â3>Iüü>pü |
| 00000090 | 1F 3F CE FA D8 3F 01 FB | B6 40 5C FB BE 41 A9 FB | ?üü? üÿ@>üâ&üü    |
| 000000A0 | C9 42 14 FC E0 43 91 FC | 95 45 06 FD 78 47 44 FD | EB üâ Cü Eÿ ÿ&Dÿ  |
| 000000B0 | 36 49 3A FD ED 4A C5 FC | 19 4C 39 FC DF 4C D3 FB | 6I:ÿiJâü I9ü&Lüü  |
| 000000C0 | A4 4D AD FB 14 4E 72 FB | 5F 4E 12 FB 36 4E 8D FA | PM-ü Nüü N ü&N ü  |
| 000000D0 | 1A 4D 53 FA AB 4B 81 FA | BC 49 02 FB 03 48 ED FB | M&Cü üüi ü Hiü    |
| 000000E0 | 00 46 17 FD D3 43 13 FE | FC 42 0C FF E4 41 0E 00 | F öüC b&B ü&â     |

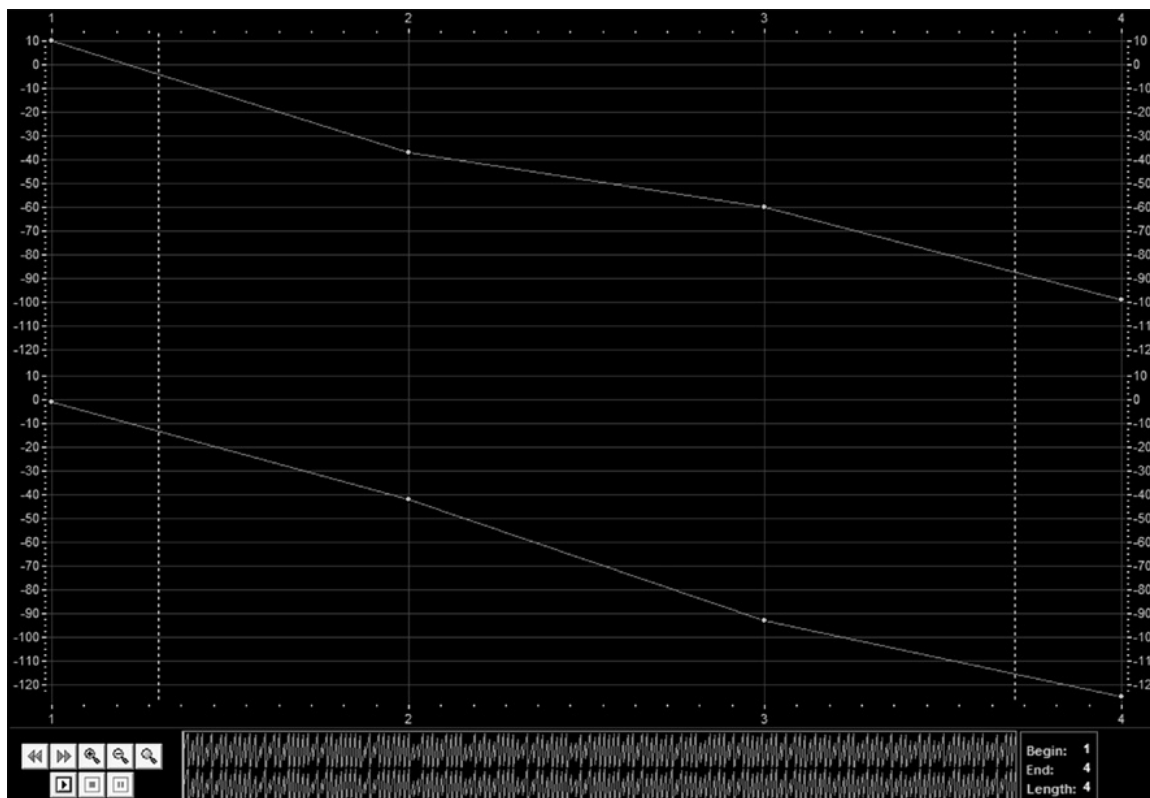
شکل ۴-۳: داده‌های نمونه برداری شده از موج اصلی

سپس، تک‌تک مقادیر داده‌ها را آزمایش می‌کنیم. چند نمونه آغازین را به کانال‌های چپ و راست تقسیم می‌کنیم، سپس مقادیر را از Little endian به big endian تبدیل می‌کنیم و مکمل ۲ را از هر ۱۶ بیت گرفته، سپس پرارزش‌ترین بیت را به عنوان بیت علامت در نظر می‌گیریم. اگر پرارزش‌ترین بیت عدد یک باشد، مقدار عدد، منفی تلقی می‌شود. ری‌کد کردن نمونه را ادامه دادیم. چهار مقدار اول به دست آمده برای کانال راست عبارت است از: ۹۳، -۴۲، -۱، -۱۲۵. اگر شکل موج را در نرم‌افزار StegoAnalyst را رندر کنیم، شکل موج تولید شده را به وسیله‌ی تمام ۳۸ ثانیه کلیپ موسیقی خواهیم دید (شکل ۴-۴).



شکل ۴-۴: فایل صوتی موج با کانال‌های چپ (بالا) و راست (پایین) برای تمام نمونه‌های گرفته‌شده

با انتخاب فقط بخشی از شکل موج (مثلاً مقادیر ۱ تا ۴) (شکل ۴-۵) مشاهده می‌کنیم که مقادیر نمونه‌های جداگانه، با مقادیری که از تبدیل دستی داده‌های خام به دست آوریم، برابری می‌کند.



شکل ۴-۵: فایل صوتی با کانال‌های چپ (بالا) و راست (پایین) برای چهار نمونه

## پنهان‌سازی داده‌ها در فایل‌های Wav

اکنون که درک خوبی از ساختار فایل Wav پیدا کردیم و مفاهیم پایه‌ی تئوری نایکویست را درک کردیم و این مطلب که چگونه این تئوری توانست مولد PCM باشد را فهمیدیم و دیدیم که چگونه مقادیر هر نمونه، به شکل اعداد صحیح ۱۶ بیتی علامت‌دار ذخیره می‌شوند، آماده‌ایم تا داده‌ها را در فایل WAV ذخیره کنیم. همان‌گونه که ممکن است حدس زده باشید، می‌توانیم از کم‌ارزش‌ترین بیت مقادیر صحیح ۱۶ بیتی، برای کد کردن پیام پنهان استفاده کنیم. در شکل ۴-۵ اولین ۸ عدد مربوط به نمونه‌های کانال راست در فایل WAV را استخراج و از آن برای پنهان‌سازی داده‌ها استفاده می‌کنیم، سپس مقادیر نمونه را از little endian به big endian تبدیل و کم‌ارزش‌ترین بیت را با بیت‌های کد اسکی حرف بزرگ A جایگزین می‌کنیم. نتایج به دست آمده به دو نکته مهم اشاره می‌کند:

- (۱) جایگزینی یک کد اسکی ۸ بیتی، به ۸ نمونه از داده‌ها در فایل Wav نیاز دارد.
- (۲) توجه کنید، در این مثال فقط ۵ بیت از ۸ بیت تغییر کرده است؛ ۳ بیت دیگر همان مقادیر پیشین می‌باشد؛ بنابراین جایگزینی، هیچ تغییر قابل پیش‌بینی‌ای بر روی LSB ندارد. وقتی داده‌ها به شکل بختانه باشند (اغلب نرم‌افزارهای پنهان‌سازی داده، نخست داده‌ها را رمزنگاری

نموده سپس اقدام به جاسازی آن‌ها در فایل حامل می‌کنند؛ در نتیجه اعداد به صورت بختانه به نظر می‌رسند)، برای جایگزینی‌های LSB معمولی، نرخ دگرگونی بیت‌ها حداکثر ۵۰٪ است. بنابراین اگر بخواهید ۸ هزار بیت را پنهان کنیم، می‌بایست حدود ۳۲ هزار بیت را تغییر دهیم، نه همه ۵۶ هزار بیت را (شکل ۴-۶).

| Offset   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | A  | B  | C  | D  | E  | F  |                |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------------|
| 00000000 | 52 | 49 | 46 | 46 | 28 | 5C | 67 | 00 | 57 | 41 | 56 | 45 | 66 | 6D | 74 | 20 | IFF(\g WAVEfmt |
| 00000010 | 10 | 00 | 00 | 00 | 01 | 00 | 02 | 00 | 44 | AC | 00 | 00 | 10 | B1 | 02 | 00 | D- ±           |
| 00000020 | 04 | 00 | 10 | 00 | 64 | 61 | 74 | 61 | 04 | 5C | 67 | 00 | 00 | 00 | 00 | 00 | data \g        |
| 00000030 | 0A | 00 | FF | FF | DB | FF | D6 | FF | C4 | FF | A3 | FF | 9D | FF | 83 | FF | yyÜyÖyÄyÿy ÿÿ  |
| 00000040 | 2F | FF | 8C | FF | D3 | FE | E5 | FF | AB | FE | 58 | 00 | ED | FE | 8D | 00 | /ÿÿÖpÿÿ«pX ip  |

|      |                     |        |                          |
|------|---------------------|--------|--------------------------|
|      | L                   | ASCII  | L                        |
|      | S                   | Hex 41 | S                        |
|      | B                   | 'A'    | B                        |
| -1   | 1111-1111-1111-1111 | 0      | 0 1111-1111-1111-1110    |
| -42  | 1111-1111-1101-0110 | 1      | -43 1111-1111-1101-0111  |
| -93  | 1111-1111-1010-0011 | 0      | -92 1111-1111-1010-0010  |
| -125 | 1111-1111-1000-0011 | 0      | -124 1111-1111-1000-0010 |
| -116 | 1111-1111-1000-1100 | 0      | -116 1111-1111-1000-1100 |
| -27  | 1111-1111-1110-0101 | 0      | -26 1111-1111-1110-0100  |
| 88   | 0000-0000-0101-1000 | 0      | 88 0000-0000-0101-1000   |
| 141  | 0000-0000-1000-1101 | 1      | 141 0000-0000-1000-1101  |

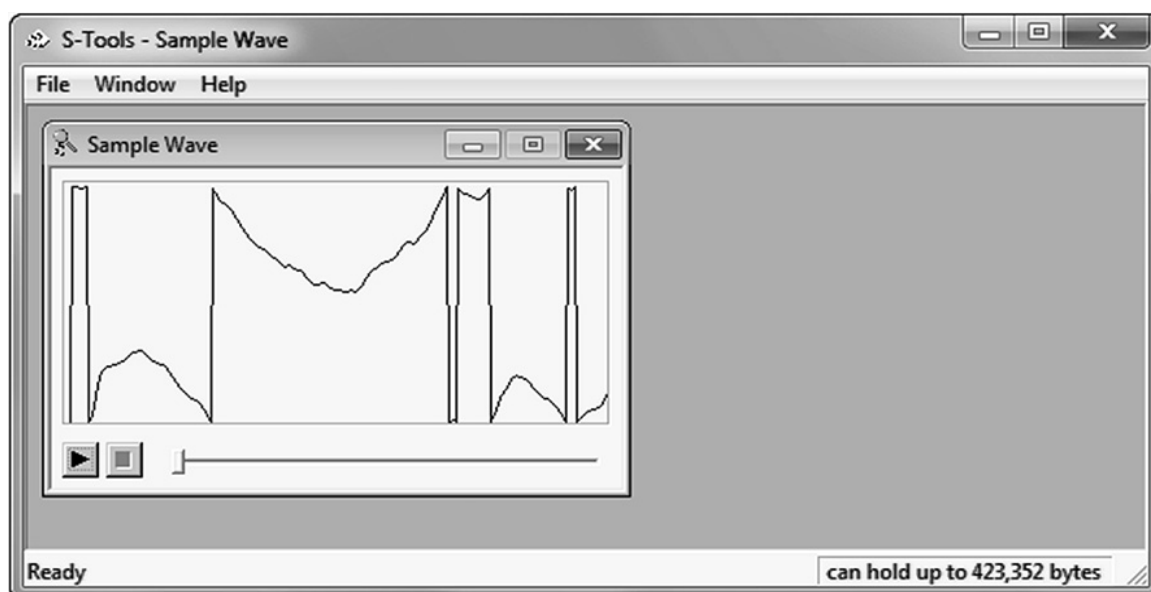
  

شکل ۴-۶: جایگزینی مقدار کم‌ارزش‌ترین بیت نمونه‌ها

حالا اجازه دهید کاربرد استتار فایل صوتی را به صورت عملی آزمایش کنیم. قصد داریم از نرم‌افزار S-tools نسخه ۴ استفاده کنیم. نخستین گام، انتخاب فایل WAV به عنوان فایل حامل است. در شکل ۴-۷ نمایش گرافیکی شکل موج فایل انتخاب شده را مشاهده می‌نمایید. دقت کنید که در سمت راست پایین صفحه، نرم‌افزار Stools حداکثر فضای خالی قابل استفاده در پنهان‌سازی را نشان می‌دهد که در این فایل برابر با ۴۲۳۳۵۲ بایت است. این عدد مقدار فضای موجود پس از فشرده‌سازی را نشان داده و تعداد کل بایت‌هایی است که می‌توان در این فایل پنهان نمود. به خاطر دارید که پیش‌تر مجموع مقادیر نمونه برای هر دو کانال را عدد ۳۳۸۶۸۸۲ محاسبه کردیم. بنابراین، اگر این عدد را بر ۸ (تعداد نمونه‌های مورد نیاز برای کد کردن هر ۸ بیت از یک بایت) تقسیم کنیم، عدد ۴۲۳۳۶۰ به دست می‌آید. زیرا ممکن

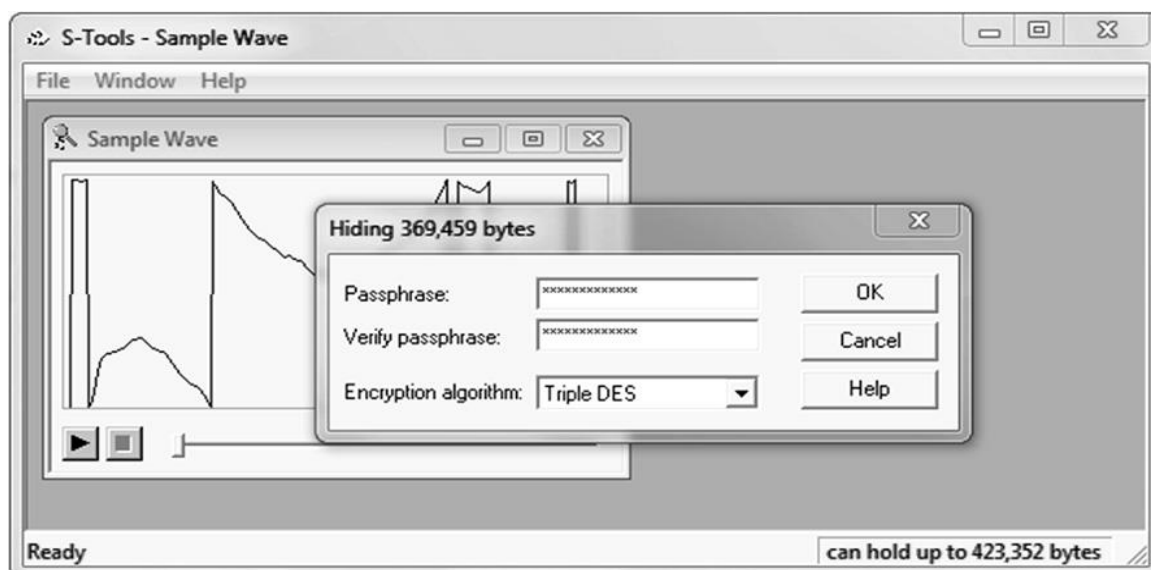


است ۸ مقدار ۱۶ بیتی را دست نخورده نگهداریم، تا نرم‌افزار S-TOOLS از این مقادیر برای اهداف خود استفاده کند.

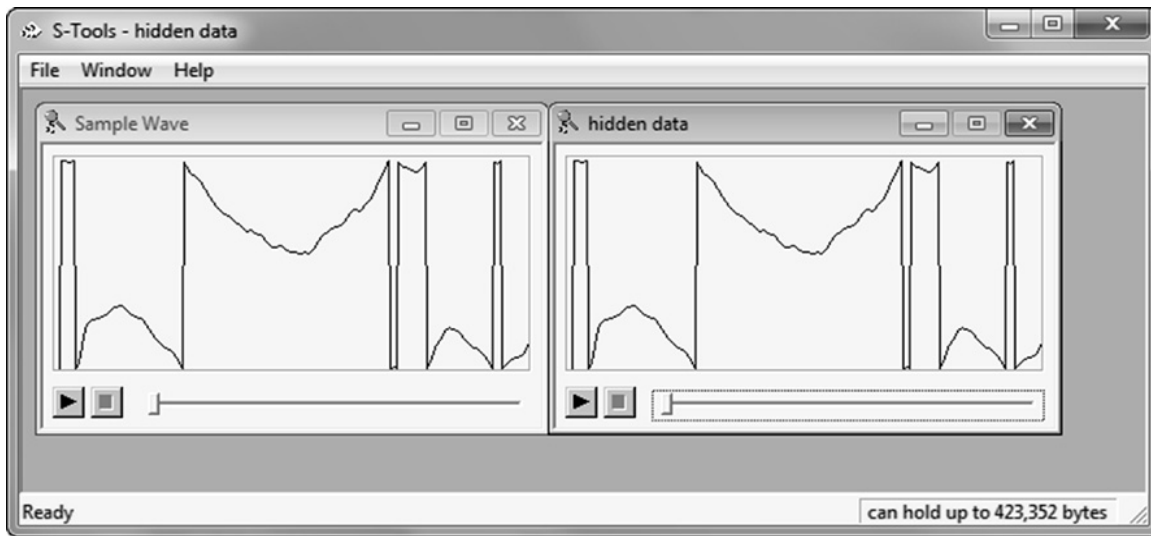


شکل ۴-۷: STTOOLS v 4.0 با نمونه آماده برای پنهان‌سازی داده‌ها.

همان‌گونه که در شکل ۴-۸ مشاهده می‌کنید، پیش از پنهان‌سازی داده‌ها، S-TOOLS قادر به رمزنگاری بر پایه‌ی گذر واژه است. شما می‌توانید از الگوریتم‌های گوناگونی برای رمزنگاری استفاده کنید. در این مثال از Trip DES استفاده کرده‌ایم. وقتی که پنهان‌سازی داده‌ها به پایان رسید، نرم‌افزار S-TOOLS مانند شکل ۴-۹ شکل موج اولیه و موج حاصل پس از پنهان‌سازی را نمایش می‌دهد. با استفاده از این شکل، بررسی سریعی از این که در خلال جاسازی داده‌ها، شکل موجی حاصل بیش‌ترین همخوانی را با شکل موج اصلی حفظ کرده است، امکان‌پذیر می‌شود.

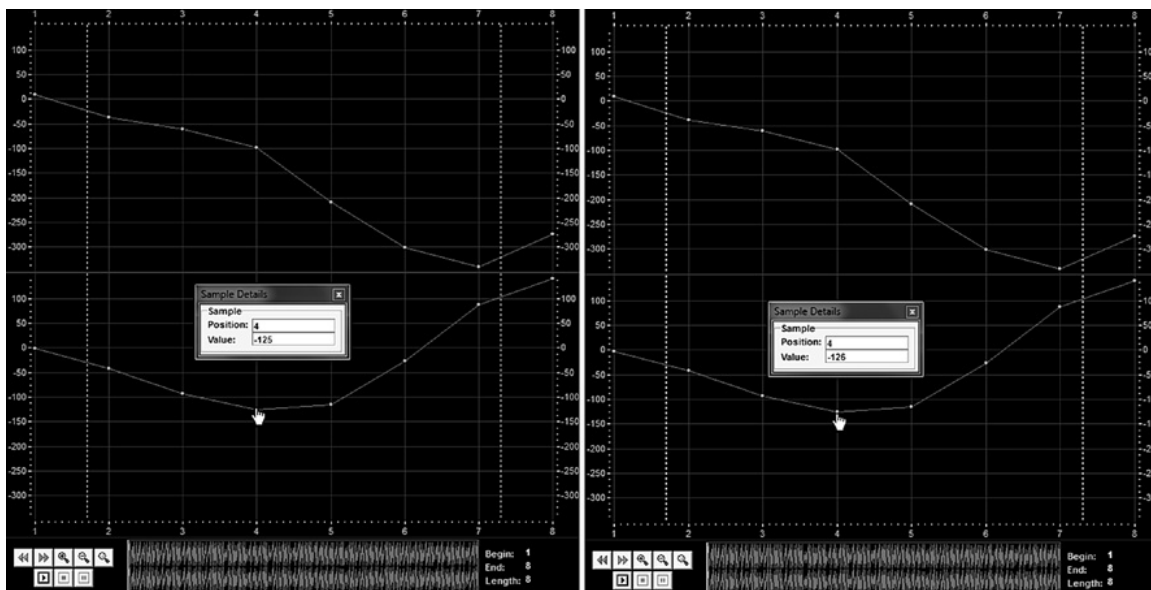


شکل ۴-۸: STOOLS v 4.0 و شکل موج حاصل از رمزنگاری به روش، Triple DES.



شکل ۴-۹: STOOLS v 4.0 مقایسه دو شکل موج پس و پیش از پنهان سازی داده‌ها

اکنون اجازه دهید شکل موج را پیش و پس از جاسازی داده‌ها و دگرگونی‌های ناشی از آن با دقت بیشتری بررسی کنیم. در شکل ۴-۱۰ برای بررسی تغییرات انجام شده در داده‌های اصلی، نگاه دقیقی به مقدار ۸ نمونه اولیه داده‌ها می‌اندازیم. مقادیر ذخیره شده برای سومین نمونه در شکل موج اصلی، مقدار ۱۲۵- بوده که به ۱۲۶- تغییر یافته است و این نشان دهنده جایگزینی مقدار کم‌ارزش‌ترین بیت با داده پنهانی می‌باشد.



Sample Wave Original

Data Hiding Wave

شکل ۴-۱۰: تحلیل شکل موج اصلی و شکل موج حاوی داده‌های پنهان

در پایان به وسیله نرم افزار Hex dump می توانیم داده های هدر<sup>۱</sup> و ۱۶ مقدار نمونه اول از هر دو کانال چپ و راست در شکل ۴-۱۱ را هم مشاهده نماییم. با نگاهی دقیق تر، تغییرات ایجاد شده به وسیله S-TOOLS در کم ارزش ترین بیت را می توان تشخیص داد.

| Offset   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 |                 |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------------|
| 00000000 | 52 | 49 | 46 | 46 | 28 | 5C | 67 | 00 | 57 | 41 | 56 | 45 | 66 | 6D | 74 | 20 | RIFF(\g WAVEfmt |
| 00000016 | 10 | 00 | 00 | 00 | 01 | 00 | 02 | 00 | 44 | AC | 00 | 00 | 10 | B1 | 02 | 00 | D~ ±            |
| 00000032 | 04 | 00 | 10 | 00 | 64 | 61 | 74 | 61 | 04 | 5C | 67 | 00 | 00 | 00 | 00 | 00 | data \g         |
| 00000048 | 0A | 00 | FF | FF | DB | FF | D6 | FF | C4 | FF | A3 | FF | 9D | FF | 83 | FF | yyÜyÖyÄyËy yly  |
| 00000064 | 2F | FF | 8C | FF | D3 | FE | E5 | FF | AB | FE | 58 | 00 | ED | FE | 8D | 00 | /ylyÖpây«pX ip  |

Sample Wave Original Header and Start of Sample Data Values

| Offset   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | A  | B  | C  | D  | E  | F  |                 |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------------|
| 00000000 | 52 | 49 | 46 | 46 | 28 | 5C | 67 | 00 | 57 | 41 | 56 | 45 | 66 | 6D | 74 | 20 | RIFF(\g WAVEfmt |
| 00000010 | 10 | 00 | 00 | 00 | 01 | 00 | 02 | 00 | 44 | AC | 00 | 00 | 10 | B1 | 02 | 00 | D~ ±            |
| 00000020 | 04 | 00 | 10 | 00 | 64 | 61 | 74 | 61 | 04 | 5C | 67 | 00 | 00 | 00 | 00 | 00 | data \g         |
| 00000030 | 0A | 00 | FE | FF | DA | FF | D6 | FF | C4 | FF | A3 | FF | 9D | FF | 82 | FF | pyÜyÖyÄyËy yly  |
| 00000040 | 2E | FF | 8D | FF | D3 | FE | E5 | FF | AB | FE | 58 | 00 | ED | FE | 8D | 00 | .y yÖpây«pX ip  |

Sample Wave Data Hiding Example Header and Start of Sample Data Values

شکل ۴-۱۱: تحلیل شکل موج اصلی و شکل موج حاوی داده های پنهان، در نمایش مبنای ۱۶

## تحلیل استتار به شیوه جاسازی داده ها در کم ارزش ترین بیت در فایل Wav

تغییر کم ارزش ترین بیت در فایل های صوتی، شیوه پنهان سازی عالی و غیر قابل تشخیصی را ارائه می کند. به عبارت دیگر، در گوش دادن به صوت اصلی و صوت تغییر یافته، حتی حساس ترین گوش ها هم توانایی تشخیص تفاوت های بین این دو را ندارند. فقط در صورتی که فایل اصلی را در اختیار داشته باشیم، می توانیم تفاوت ها را مشخص کنیم. با توجه به تفاوت های ایجاد شده بین دو موج، می توان برداشت کرد که پنهان سازی داده های در کم ارزش ترین بیت انجام شده است. با این وجود در اکثر موارد، فقط فایل صوتی تغییر یافته را برای آزمایش در اختیار داریم. در این صورت کشف داده هایی به روش جاسازی داده ها در کم ارزش ترین بیت فایل اصلی انجام شده و می بایست بدون کمک گرفتن از فایل اصلی انجام شود. هدف اساسی، تعیین مقادیر تغییر یافته کم ارزش ترین بیت در فایل اصلی که حاوی اطلاعات است می باشد، نه تغییراتی که بر اثر نویزهای تصادفی در کم ارزش ترین بیت به وجود آمده است. به عنوان مثال، اگر برای پنهان سازی مجبور به تغییر مقادیر کم ارزش ترین بیت در هر نمونه باشیم، باید این تغییر به شکلی انجام

<sup>۱</sup> header

شود که دست‌کم شکلی از موسیقی اصلی هم قابل باز تولید باشد. بنابراین، کلید تشخیص فایل حاوی داده‌های جاسازی شده تعیین این نکته است که آیا مقادیر کم‌ارزش‌ترین بیت‌های موجود در فایل حاوی اطلاعات است یا فقط نویزهای تصادفی دارد. روش رایج تشخیص این دو، تخمین قابلیت فشرده‌سازی جریان بیتی حاصل است. در مورد فایل‌های صوتی، نخست کم‌ارزش‌ترین بیت هر کانال را مشخص و سپس تحلیل آماری متنی را انجام می‌دهیم.

یک راه چیره شدن بر احتمال کشف داده‌های پنهان در فایل‌های صوتی به روش تغییر کم‌ارزش‌ترین بیت، استفاده از Warden فعال می‌باشد. Warden، مقدار کم‌ارزش‌ترین بیت تعداد خاصی از نمونه‌ها را صفر می‌کند. این کار باعث ایجاد تغییرات محسوسی در زمان پخش فایل صوتی نمی‌شود، بلکه فقط رندر کانال را بی‌اثر می‌کند.

### جاسازی داده‌ها در فایل صوتی به روش پیشرفته

با گذشت زمان، پیشرفت روش‌های جاسازی داده‌ها در فایل‌های صوتی به کندی پیش می‌رود. حتی امروزه، برنامه‌های استتار انگشت‌شماری وجود دارند که فایل‌های صوتی فشرده مانند AAC و MP3 را پشتیبانی کند. شاخص‌ترین آن‌ها، برنامه‌ی MP3steg است که روش ویژه‌ای از کوانتیزه<sup>۱</sup> را به کار می‌برد و سپس اقدام به جاسازی داده‌ها در بلوک‌های توازن داده‌ها در فایل MP3 می‌نماید. انکدر MP3 فایل WAV را به عنوان ورودی به همراه پایه‌بار پذیرفته و یک فایل MP3 را با داده‌های جاسازی شده به عنوان خروجی تولید می‌کند. مثلاً فایل صوتی‌ای با حجم Mb6 که تمام طیف‌های صوتی را شامل شود، تنها می‌تواند Mb6 یا کمتر پایه‌بار داشته باشد، یعنی چیزی حدود ۱٪ (شکل ۴-۱۲).

```

MP3StegoEncoder 1.1.15
See README file for copyright info
USAGE : encode [options] <infile> <outfile>
OPTIONS : -h          this help message
          -b <bitrate> set the bitrate, default 128kbit
          -c          set copyright flag, default off
          -o          set original flag, default off
          -E <filename> name of the file to be hidden
          -P <text>    passphrase used for embedding

MP3StegoEncoder 1.1.15
See README file for copyright info
Microsoft RIFF, WAVE audio, PCM, stereo 44100Hz 16bit, Length: 0: 0:38
MPEG-I layer III, stereo Psychoacoustic Model: AT&T
Bitrate=128 kbps De-emphasis: none CRC: off
Encoding "inwave.wav" to "sample.mp3"
Hiding "simpleinsert.txt"
Enter a passphrase:
Confirm your passphrase:
[Frame 1470 of 1470] (100.00%) Finished in 0: 0:16

```

شکل ۴-۱۲: MP3Stego Command Line MP3 Encoder دستور متنی و سویچ‌های آن

### چکیده‌ی بخش پنهان‌سازی داده‌ها در فایل‌های صوتی

پیشرفت پنهان‌سازی داده‌ها در فایل‌های صوتی به کندی پیش می‌رود. با گذر از کم‌ارزش‌ترین بیت که توانایی‌های زیادی را در جاسازی داده‌ها بدون امکان تشخیص شنیداری را فراهم می‌کند به روش‌هایی بر پایه سیستم شنیداری انسان و کدگذاری MP3 می‌رسیم که حجم کمی از داده‌ها را جاسازی کرده، اما تشخیص و کشف آن‌ها را بسیار مشکل می‌کند. امروزه تقاضای بیشتر روی کاربرد روش‌های پنهان‌سازی داده‌ای است که بر روی گوشی‌های هوشمند کار کند و امکان استتار/پنهان‌سازی داده‌ها را به صورت همزمان با ارسال جریان داده فراهم نماید.

### پنهان‌سازی داده‌ها در فایل‌های ویدئویی دیجیتال

پنهان‌سازی و استتار داده‌ها مبتنی بر داده‌های ویدئویی دیجیتال، پتانسیل بسیار بالایی در کسوت کانال پوششی دارد و این بیشتر به خاطر اندازه‌ی بزرگ‌تر این فایل‌ها و تعداد زیاد فایل‌های ویدئویی و همچنین کاربرد گسترده‌ی آن‌ها است که امروزه در اینترنت و در فضای Cloud جریان دارد. این فایل‌ها، هدف امروزی و آتی کسانی است که می‌خواهند ارتباطاتشان را پنهان کنند. ویدئو دیجیتال دو شکل اصلی فشرده و غیر فشرده دارد. رایج‌ترین شکل فایل‌های فشرده Mpeg است. Mpeg با حذف افزودگی‌های موجود در فضا و زمان در زمان کدگذاری داده‌ها به نرخ بالای فشرده‌سازی دست می‌یابد. جریان بیتی

ویدیویی به دست آمده با این روش متشکل از کدهایی با طول متغیر است که تصویر را به وسیله‌ی روش تقسیم آن به بخش‌های زیاد نمایش می‌دهند.

## چکیده

پنهان‌سازی اطلاعات در فایل‌های ویدئویی غیرفشرده مانند AVI و فشرده مانند MPEGx نه تنها امروزه امکان‌پذیر است، بلکه می‌تواند منبع قابل توجهی برای انتقال پیوسته‌ی داده‌های جاسازی شده باشد. با روش‌های فعلی و آتی در زمینه‌ی تصحیح خطا و افزونگی داده‌ها، امکان بقای داده‌های پنهان شده، حتی در شرایط نویزی خطوط هم وجود دارد. بی‌شک روند پنهان‌سازی اطلاعات، سطح جدیدی از تهدید امنیت را نیز پیش روی ما می‌گذارد. در نتیجه تحلیلگر استتار، وظیفه دشوار شناسایی، یا دست‌کم محدود نمودن کانال‌های را به عهده دارد. هدف او جلوگیری از ارتباطات پنهانی، پیشگیری از ارسال فایل با مالکیت معنوی یا استفاده از کانال‌های جریان برای رساندن دستور و کنترل سایر کدهای مخربی است که می‌تواند تهدیدهای مداوم در سطح پیشرفته را سرعت بخشیده یا آسان نماید.

## فصل پنجم

### پنهان سازی داده ها در ابزارهای همراه اندرویدی

#### پنهان سازی داده ها در ابزارهای اندرویدی به وسیله نرم افزار **Img Hid** و **APP آشکار سازی آن**

نرم افزار Image Hide (ImgHid) امکان تعریف تلفن مجازی در داخل تلفن واقعی را فراهم می کند. این نرم افزار مثل بسیاری از APP اندرویدی پنهان سازی داده ها در تصاویر، ترجیح می دهد از فایل های jpeg استفاده نماید و در این مورد با APP آیفون که ترجیح می دهند از فایل های PNG برای اهداف پنهان سازی استفاده کنند، تفاوت دارد. از آنجایی که توسعه APP مبتنی بر جاوا در اندروید، همگانی و فراگیر است و با توجه به فراوانی الگوریتم های گوناگون مبتنی بر قالب Jpeg در جاوا، این گزینه را به گزینه ی برتر در پنهان سازی تبدیل نموده است.



شکل ۵-۱: عکس حاملی که عکس سری را در دل خود جای می‌دهد.

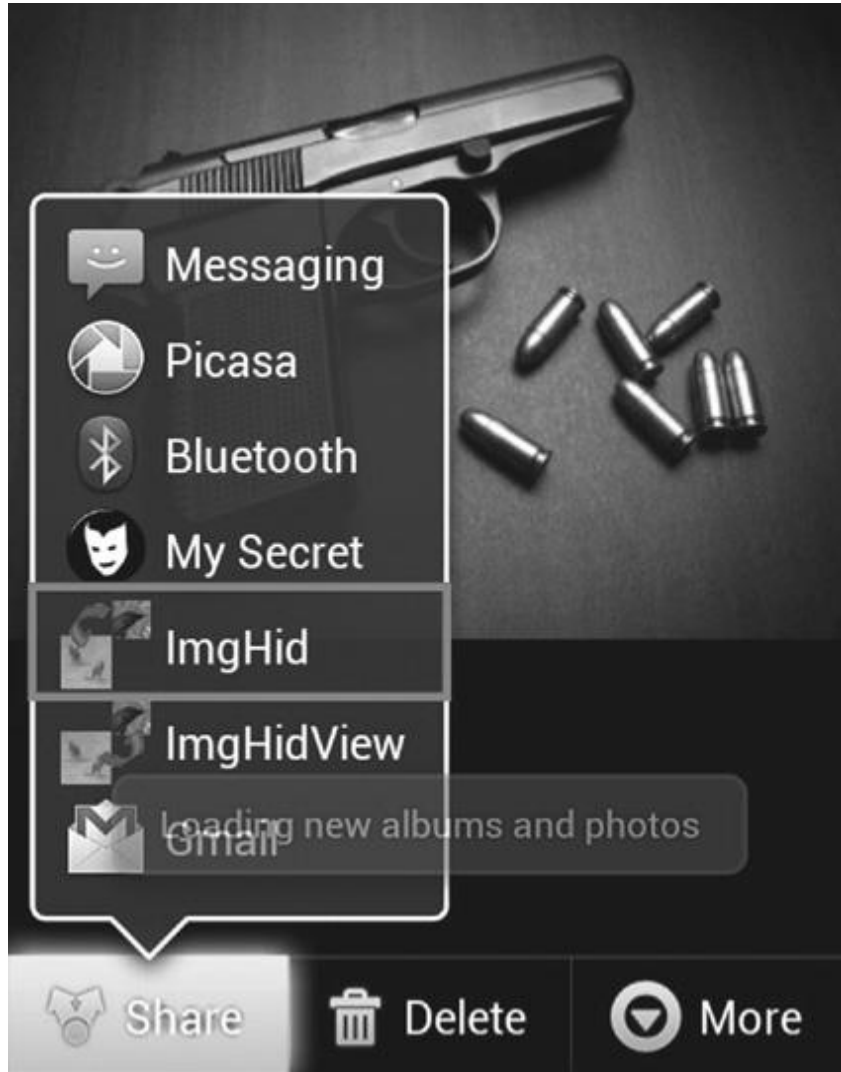


شکل ۵-۲: تصویر انتخاب شده به عنوان عکس سری که باید در عکس دیگری جاسازی شود.

| Image Hide and Reveal Details |                   |
|-------------------------------|-------------------|
| Application Name              | ImgHid and Reveal |
| Developer/creator             | actfor-j          |
| Carrier format                | JPEG              |
| Last release                  | July 2011         |

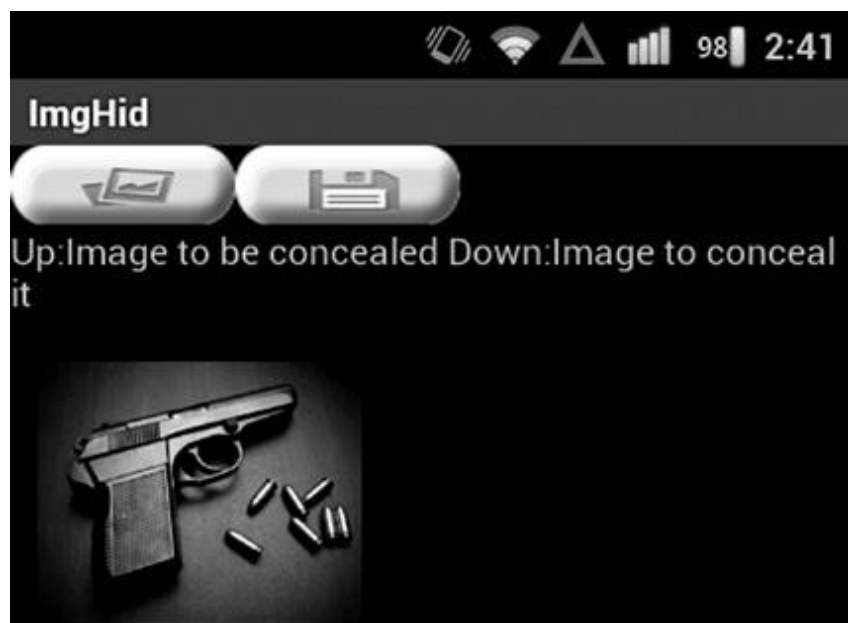


مثل بسیاری از APP های اندروید و آیفون، رابط کاربر و عملکرد این نرم افزار ساده است. رابط کاربری Img Hid را در شکل ۳-۵ مشاهده می کنید. با انتخاب نمایه Img Hid فرایند پنهان سازی داده ها را شروع می کنیم.



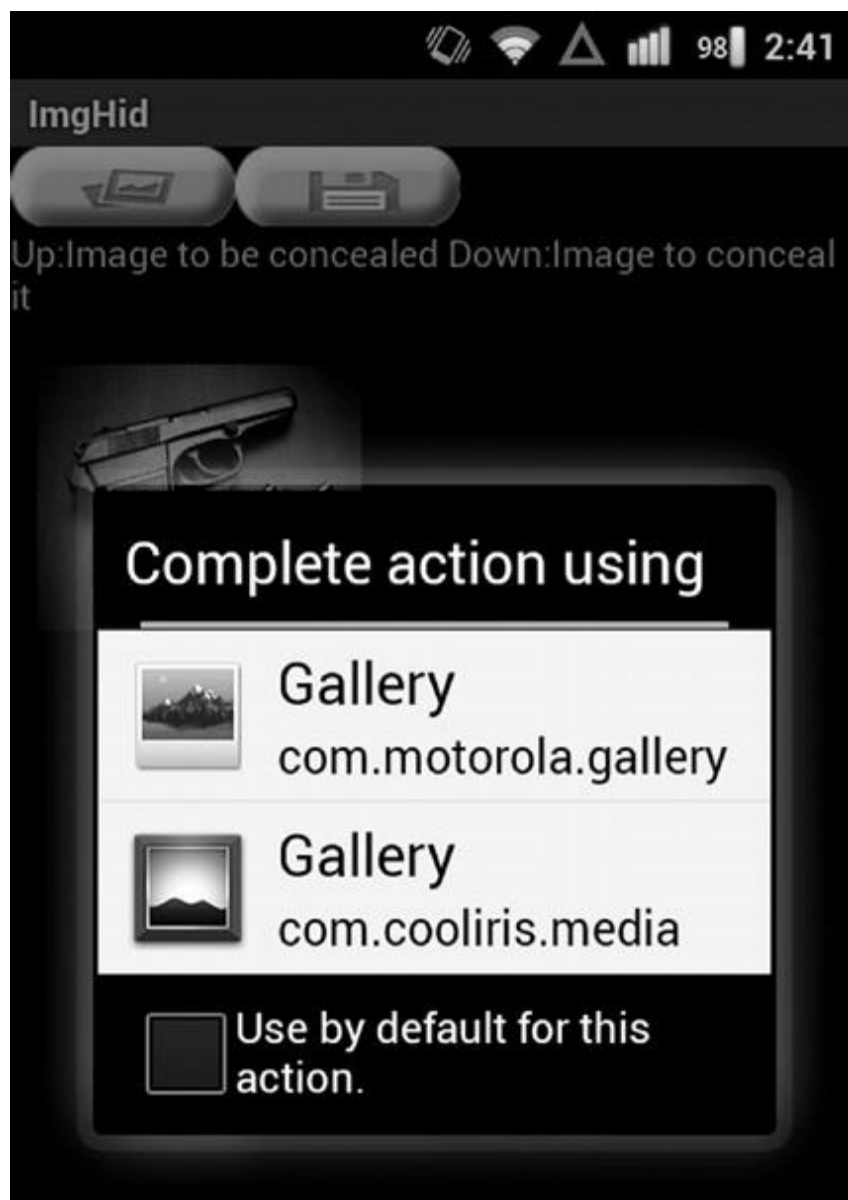
شکل ۳-۵: رابط کاربری نرم افزار Img Hid

در Img Hid نخست باید مشخص کنید چه عکس یا تصویری را می خواهید پنهان کنید. به عبارت دیگر نخست باید عکس سری را انتخاب نماییم. در شکل ۴-۵ مشاهده می کنید که عکس تفنگ و مهمات آن را به عنوان عکسی که می خواهیم پنهان کنیم انتخاب کرده ایم.



شکل ۵-۴: انتخاب عکس سری که می‌خواهید در دل عکس دیگری جای دهید

دوباره با استفاده از گزینه Gallery عکس جغد برفی را به عنوان عکس پوشش برای داده‌های پنهان استفاده می‌کنیم. (عکس ۵-۵) هر دو عکس اسلحه و مهمات و جغد برفی را در شکل ۵-۶ مشاهده می‌کنید.



شکل ۵-۵: گالری نرم‌افزار Img Hid



شکل ۵-۶: عکس سری (بالا) و عکس حامل (پایین)

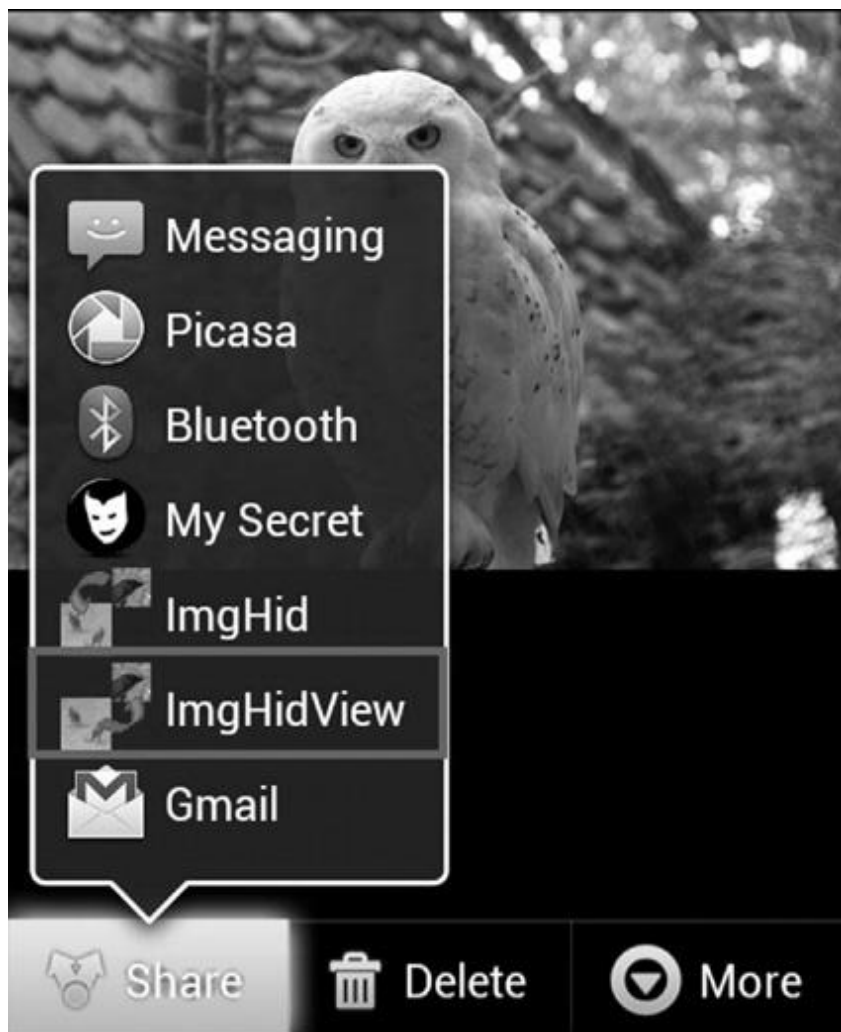
در این مرحله آماده‌ی ترکیب عکس‌ها هستیم. این نرم‌افزار پنهان‌سازی عکس سری (اسلحه و مهمات) را در عکس حامل (جغد برفی) با موفقیت به پایان می‌رساند (شکل ۵-۷).



شکل ۵-۷: پایان موفقیت آمیز پنهان‌سازی عکس سری در عکس حامل

اکنون با استفاده از مرورگر اینترنت که امکان استفاده از خدمات ایمیل برای ارسال عکس حاصل حاوی عکس سری را فراهم می‌نماید، اقدام به ارسال آن به مقصد می‌نماییم (شکل ۵-۸).

حالا به تحلیل نتیجه می‌پردازیم.



شکل ۵-۸: عکس حاصل از پنهان‌سازی

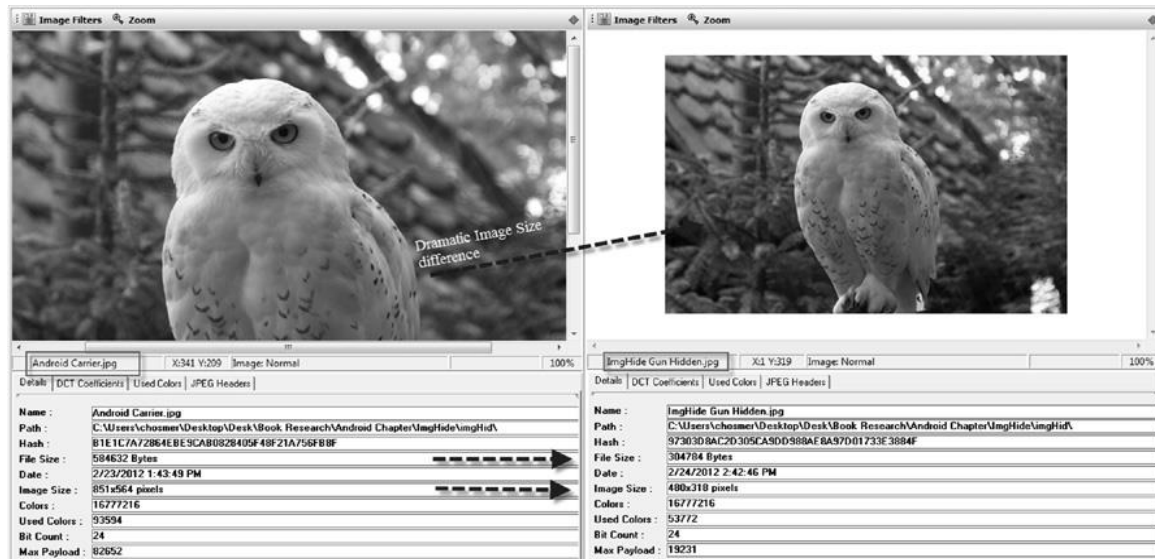
### تحلیل نتیجه حاصل از پنهان‌سازی داده در Img Hid

برای آزمودن روش و چگونگی عملکرد اغواگرانه‌ی پنهان‌سازی عکسی در داخل عکس دیگر به وسیله‌ی نرم‌افزار Img Hid نخست نگاهی دقیق به عکس حامل پس از پنهان‌سازی انداخته و آن را با عکس اصلی جغد برفی مقایسه می‌کنیم. در شکل ۵-۹ چند تغییر را پیش و پس از پنهان‌سازی عکس مشاهده می‌کنیم. عکس جغد برفی اولیه در سمت چپ و جغد برفی حامل عکس تفنگ و مهمات در دست راست نمایش داده شده است.

تفاوت‌ها :

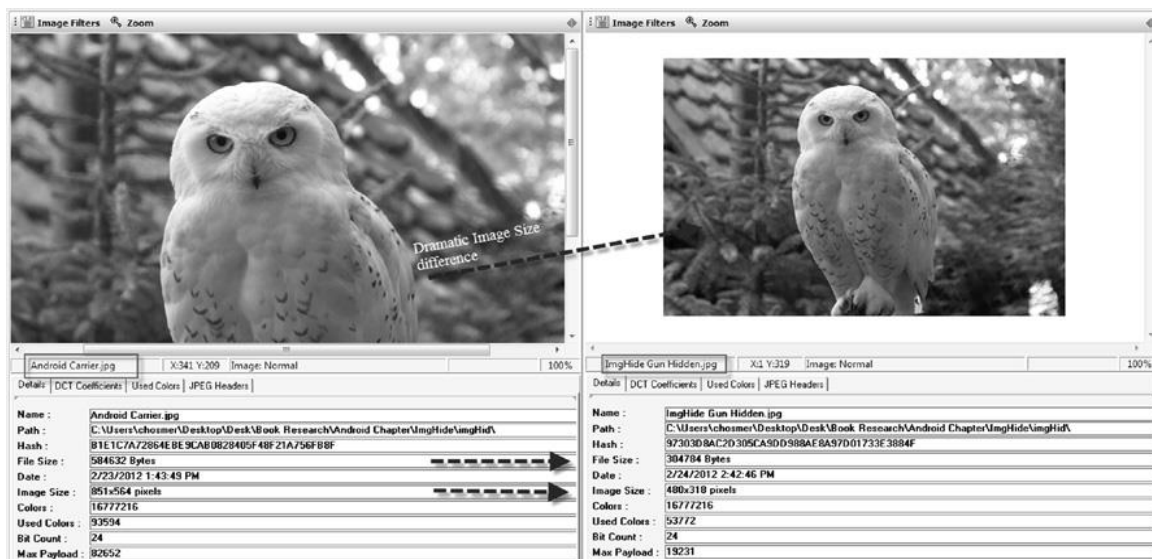
(۱) مقیاس عکس اصل به طرز چشمگیری کاهش یافته شده است.

(۲) ابعاد عکس حاصل باز هم کوچک‌تر شده است.



شکل ۵-۹: عکس حامل پیش و پس از پنهان سازی داده ها در آن

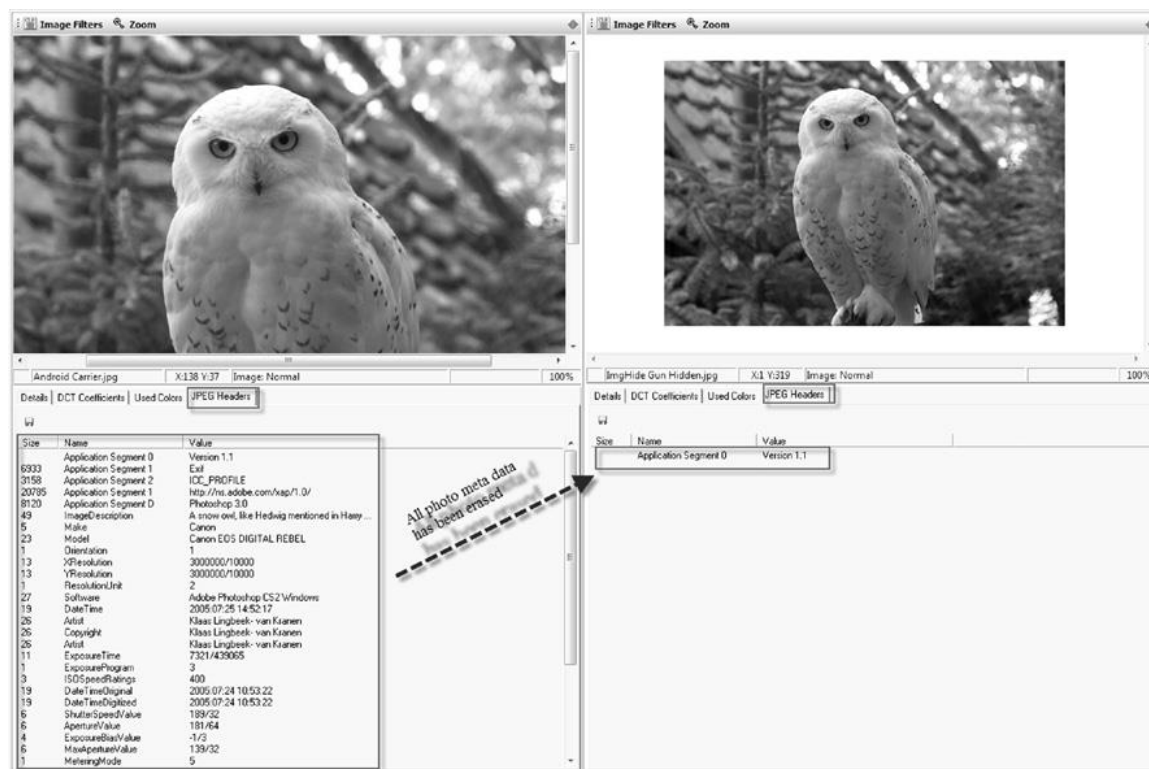
با نگاهی دقیق تر به چگونگی جاسازی داده های عکس پنهان در عکس حامل و همان گونه که در شکل ۵-۱۰ مشاهده می نمایید، تمام ابر داده های عکس حامل در عکس حاوی تصویر سری حذف شده اند.



شکل ۵-۱۰: تغییر ابر داده های فایل حامل پیش و پس از پنهان سازی داده ها

از نشانگرهای موجود در فایل Jpeg برای تشخیص داده های جاسازی شده در عکس استفاده می کنیم. مشاهده می گردد (عکس ۵-۱۱) که کد نشانگر پایان تصویر FF D ۹ عکس اصلی در سمت چپ در آخر فایل است. این نشانگر، انتهای داده ها را مشخص کرده و باید در آخر فایل Jpeg باشد. همان گونه

که مشاهده می‌کنید این نشانگر در محل خود در فایل حاوی داده‌های جاسازی شده وجود دارد، لیکن پس از آن مقدار قابل توجهی داده‌ها به فایل افزوده شده است.



شکل ۵-۱۱: عکس حامل پس و پیش از افزودن داده‌ها پس از نشانگر پایان داده در فایل Jpeg

حال با مقایسه‌ی دو فایل متوجه شدیم که روش استتار به‌کاررفته در نرم‌افزار Img Hid روش افزودن داده‌ها پس از پایان نشانگر تصویر در فایل Jpeg می‌باشد. هر دو فایل دقیقاً با عکس جغد برفی شروع می‌شوند ولی در مقدار پایه‌بار باهم فرق دارند. در شکل ۵-۱۲ دو عکس جغد برفی با پایه‌بار متفاوت را مقایسه کردیم. عکس سمت چپ (عکس اصلی) از پایه‌بار اولیه استفاده کرده و عکس سمت چپ سعی در پنهان کردن عکس اسلحه و مهمات به عنوان پایه‌بار دارد.

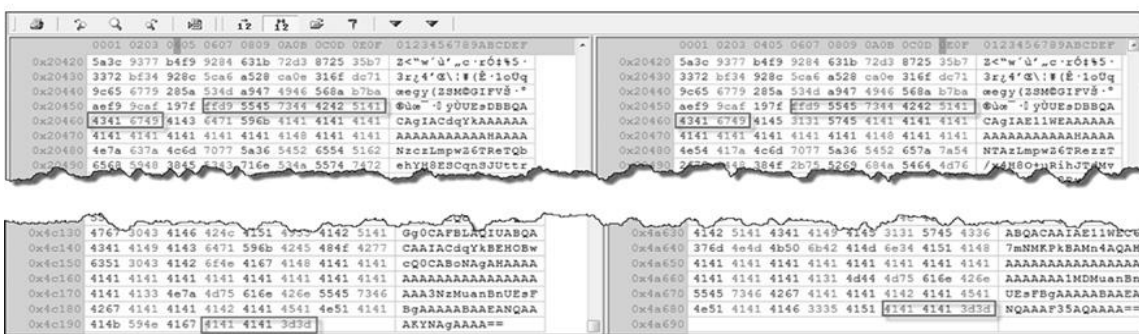




شکل ۵-۱۲: مقایسه عکس حامل و عکس با داده های جاسازی شده

با نگاه دقیق تر به محتویات دو عکس، الگوهایی آشکار می شود که نه تنها به ما اجازه می دهد تا تشخیص دهیم که داده های پنهان در این دو عکس وجود دارد یا خیر، بلکه می توان منبع پنهان سازی داده ها را هم مشخص نماییم.

همان گونه که در عکس ۵-۱۳ مشاهده می کنید داده های هر دو عکس پس از نشانگر ۹ FF D با مقدار ۵۵ ۴۹ ۶۷ ۴۳ ۴۱ ۵۱ ۴۲ ۴۲ ۴۴ ۷۳ ۴۵ شروع و با 41 41 41 41 41 D3 به پایان می رسند.



شکل ۵-۱۳: نشانگر استاندارد Jpeg در هر دو عکس حامل و عکس با داده های پنهان

الگوی تمام پایه بار عکس هایی که ما آزمایش کردیم، همین الگو بود.

محتویات داده بین آغاز و پایان نشانگر، کدهای اختصاصی مربوط به فایل عکس پنهان شده را هم شامل می‌شود. (در این مثال عکس اسلحه و مهمات) تأثیر قابل توجه در خصوص پایه‌بار، کاهش چشمگیر وضوح و ابعاد تصویر پس از پنهان‌سازی است. عکس ۵-۱۴ عکس اصلی تفنگ و مهمات و عکس بازیابی شده از جغد برفی را نشان می‌دهیم.



شکل ۵-۱۴: اندازه عکس پنهان شده پیش و پس از پنهان‌سازی

لب مطلب این است که APP اندرویدی Img Hid کارکرد آسانی داشته و ویژگی‌های پایه پنهان-سازی داده‌ها را به خوبی ارائه می‌دهد؛ به عبارت دیگر عکس با داده‌های پنهان در مشاهده‌ی معمولی، عادی به نظر می‌رسد ولی مشاهده‌ی دقیق‌تر، روش معمولی افزودن داده‌ها به آخر فایل را آشکار می‌کند. روشی که به سادگی قابل پیگیری و تشخیص است.

## APP اندروید My Secret

| Android My Secret App Details |                |
|-------------------------------|----------------|
| Application Name              | My Secret      |
| Developer/creator             | Tipspedia Ro   |
| Carrier format                | JPEG           |
| Last release                  | September 2011 |

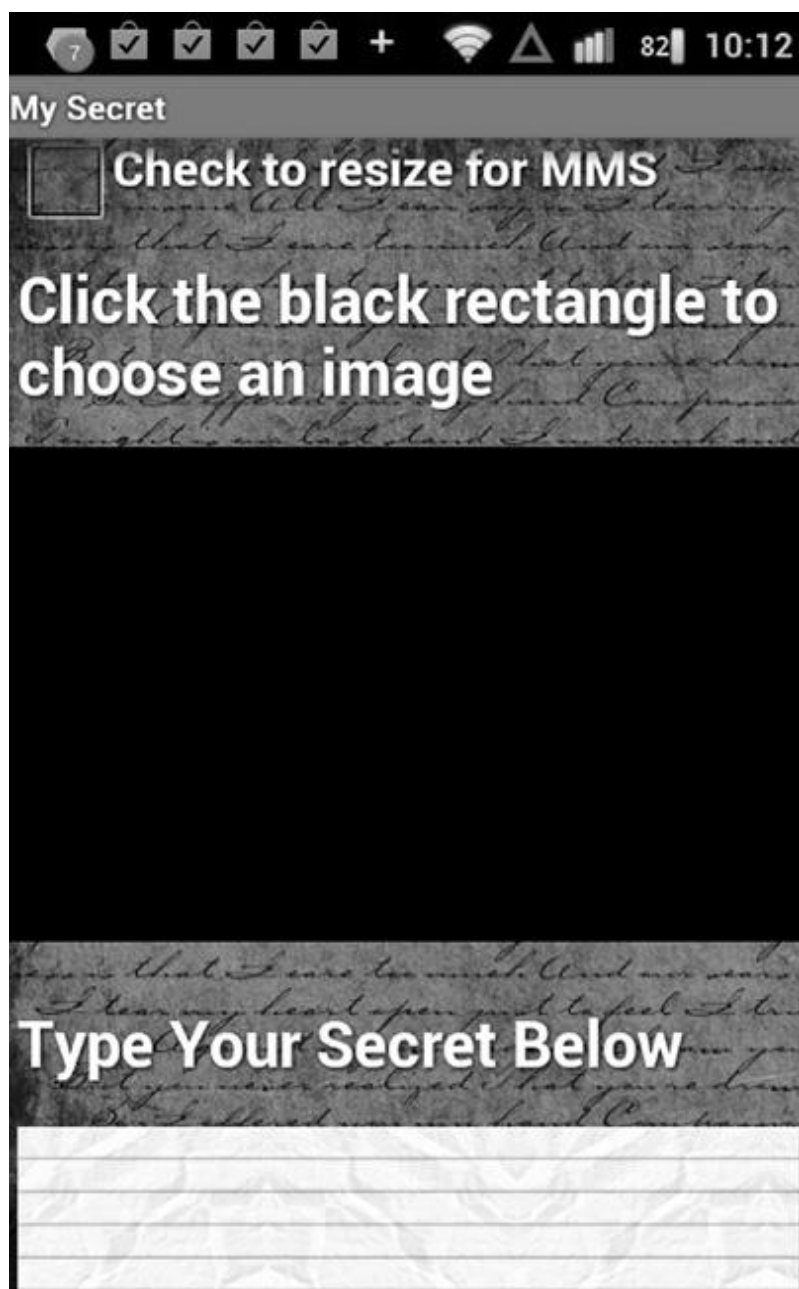
همانند سایر APP اندرویدی و آیفون، My Secret هم به راحتی قابل استفاده بوده و در نتیجه می‌توان به سرعت و پس از اجرا، به تحلیل عملکرد آن پردازیم.

صفحه مرورگر نرم افزار امکان انتخاب یا گرفتن عکس که می خواهید پنهان نماییم را می دهد (شکل ۵-۱۵).



شکل ۵-۱۵: رابط کاربر نرم افزار My Secret

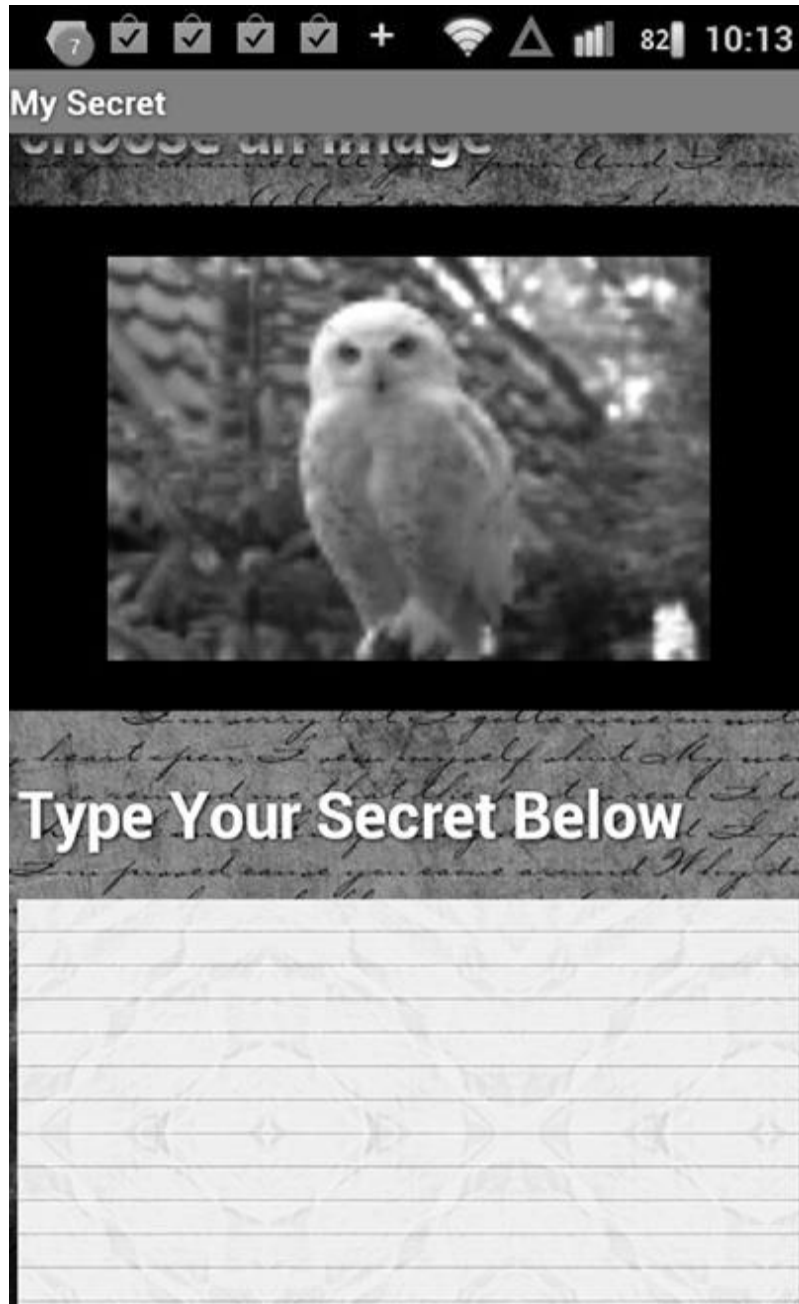
با انتخاب گزینه‌ی ایجاد، پنجره‌ای را که در شکل ۵-۱۶ مشاهده می‌کنید باز می‌شود و با کلیک در پس زمینه App فایل حامل که برای عملیات پنهان‌سازی داده‌ها نیاز داریم را انتخاب می‌کنیم.



شکل ۵-۱۶: پنجره انتخاب فایل حامل پیام پنهانی

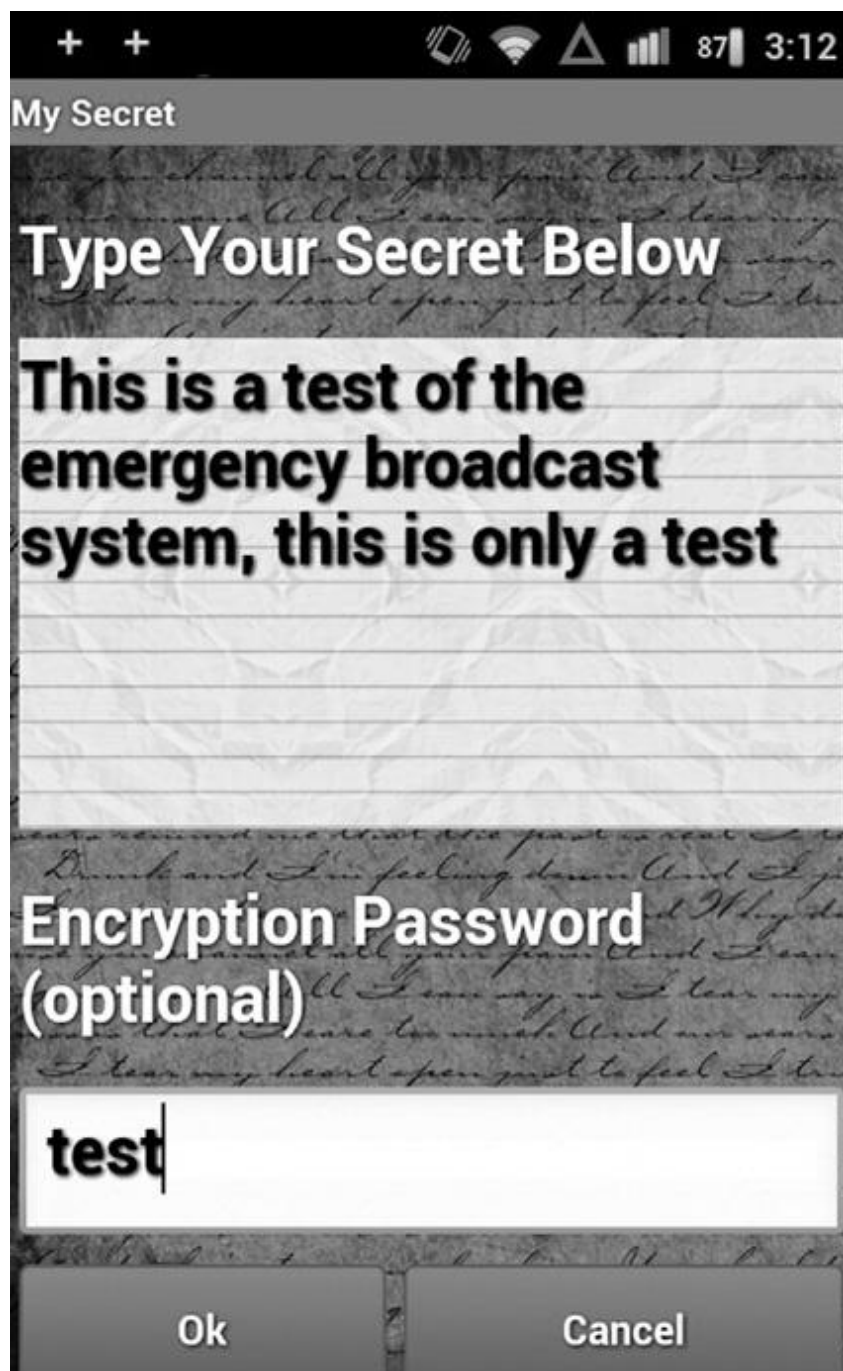
دوباره عکس جغد برفی را به عنوان فایل حامل و پوششی برای فایل سری انتخاب می‌کنیم (۵-۱۷).

(۱۷).



شکل ۵-۱۷: پیش نمایش انتخاب فایل حامل

از آنجایی که My Secret تنها امکان پنهان‌سازی متن در داخل عکس حامل را می‌دهد، پیام سری که باید انتقال یابد را تایپ می‌کنیم. در صورت تمایل می‌توانیم گذرواژه هم مشخص کنیم (شکل ۵-۱۸)



شکل ۵-۱۸: صفحه افزودن متن پیام سری

### تحلیل نتایج حاصل از پنهان‌سازی داده‌ها در My Secret

گام‌های تحلیل در My Secret App همانند راه‌های نرم‌افزار مشابه آن است. نخست نگاهی به عکس حاصل و اندازه‌ی آن می‌اندازیم. شکل ۵-۱۹ نشان می‌دهد که هر دو عکس اندازه‌ی تقریباً یکسانی دارند

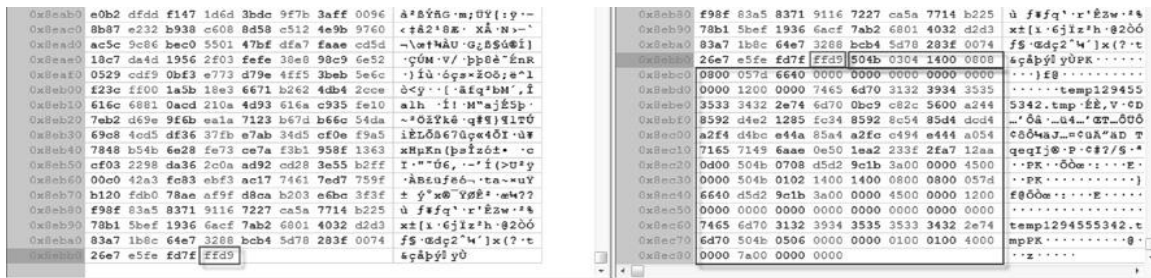
و ابعادشان دقیقاً یکسان است. به علاوه هر دو عکس نمایش یکسانی از محتویات را زیر ذره بین و پرداز زدن های مختلف و بازرسی های دقیق ارائه می دهد.



شکل ۵-۱۹: تحلیل ابعاد و چگونگی نمایش عکس، پیش و پس از افزودن پیام پنهان

در مرحله ی بعدی، ابر داده ها در سرایند هر دو فایل را بررسی می کنیم؛ تغییری مشاهده نخواهیم کرد. بنابراین حقایق زیر را می توانیم استنتاج کنیم:

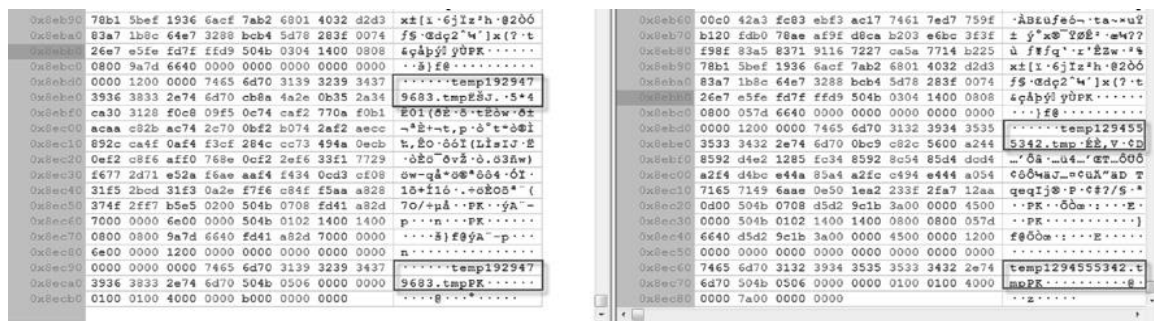
- (۱) به نظر نمی رسد فایل دوباره کد شده باشد تا DCT quantiec اعداد متفاوتی تولید کند.
  - (۲) افزودن داده به سرایند هم دور از ذهن است زیرا ابر داده های هر دو فایل مشابه هم هستند.
- آزمودن نشانگرهای دیگر فرمت Jpeg مثل EOF واقعیت را آشکار می کند. همان گونه که در شکل ۵-۲۰ مشاهده می کنید، فایل اصلی نشانگر ۹ FF D را در محل مناسب خود و در انتهای فایل دارد، لیکن عکس تغییر یافته چنین نیست و مهم تر آن که مقدار کوچکی اطلاعات به اندازه ی پیام متنی که تایپ کردیم به فایل اضافه شده است.



شکل ۵-۲۰: تغییرات نشانگر Eof پیش و پس از فشرده سازی

## خلاصه

My Secret اندرویدی پنهان‌سازی داده‌ها را به روش مناسبی که هیچ‌گونه تأثیر دیداری بر عکس با هر درجه از دقت و موشکافی نمی‌گذارد. با این حال با بررسی نشانگر معروف Jpeg می‌توان به وجود داده‌های اضافی پی برد، مانند Img Hid App امضا نرم‌افزار را هم می‌توان از داده‌های پنهان شده استخراج نمود. در شکل ۵-۲۱ دو فایل حاصل از My Secret را با هم مقایسه می‌کنیم. فایل سمت چپ شامل با گذر واژه حفاظت می‌شود در صورتی که فایل سمت راست بدون گذر واژه است. توجه کنید که در هر دو عکس متن پنهان شده با رشته‌ی مشترکی آغاز شده و به پایان می‌رسد. برای عکس سمت چپ نشانگر Temp ۱۹۲۹۴۷۹۶۸۳ و برای عکس سمت راست نشانگر Temp ۱۲۹۴۵۵۵۳۴۲ است.



شکل ۵-۲۱: تعیین نشانگرها در My Secret Stego

این اعداد در واقع اعداد صحیحی است که بیانگر زمان طی شده از مبدأ زمانی خاصی هستند. برای قابل‌فهم شدن آن‌ها اجازه دهید به شکل استاندارد نمایش (-0500 Eastern Standard Time) GMT را به ۰۱:۴۲:۲۲ ۲۰۱۱ ۰۹ Jun تبدیل کنیم. این گزینه امکان بیشتری برای تحقیقات قضایی را نیز فراهم می‌کند.



یک بار دیگر با ترکیب روش های تشخیص داده های اضافه شده پس از نشانگر، نه تنها به وجود داده های پنهان پی می بریم، بلکه می توان تشخیص داد که My Secret نرم افزاری است که این داده ها را به فایل اصلی اضافه کرده است.

شایان توجه است که بیشتر نرم افزارهایی که پنهان سازی داده ها را فایل های Jpeg انجام می دهند به روش اضافه کردن داده ها پس از نشانگرهایی که مسئول نمایش بخش داده های عکس هستند اضافه می کنند.

## نرم افزار STEGDROID

| StegoDroid Details |            |
|--------------------|------------|
| Application Name   | StegDroid  |
| Developer/creator  | Tom Medley |
| Carrier format     | .ogg Audio |
| Last release       | March 2011 |

STEGDROID برنامه ی کاربری رایگانی است که می توانید آن را از سایت Market اندروید تهیه کنید. هدف اصلی تولیدش، فشرده کردن و سپس به اشتراک گذاردن پیام های متنی کوتاه به شکل کلیپ صوتی است. این App برخلاف SMS و MMS تلاش می کند که پیام ها را در لفافه ای قرار داده تا از شر فیلتر شدن و تشخیص بگریزد. نکته جالب در مورد این APP استفاده از شیوه ی پنهان سازی موسوم به استتار انعکاسی است. استتار انعکاسی<sup>۱</sup>، همان گونه که از نامش پیداست داده ها را نزدیک نواحی ای که در حالت عادی در زمان ضبط ممکن است حالت انعکاس داشته باشند، ذخیره می کند؛ مثل انعکاس صدای حاصل از برخورد صدا با دیوار، پنجره، میز، صفحه نمایش کامپیوتر، صفحه کلید. این انعکاس ها معمولاً توسط سیستم شنیداری و مغز انسان نادیده گرفته می شوند. به استناد تحقیقی که به وسیله ی Jenkins و Martina در سال ۲۰۰۹ میلادی انجام شد، این روش استتار آستانه ی تشخیص ناپذیری را در بیشتر شرایط در نظر می گیرد. به علاوه، استتار انعکاسی چند مزیت نسبت به اشکال دیگر جاسازی داده ها در فایل های صوتی به شرح زیر دارد:

(<sup>۱</sup>) این روش در ایجاد تغییراتی که تشخیص ناپذیر و پایدار در برابر کشف و پارازیت باشد، موفق عمل می کند (JAM).

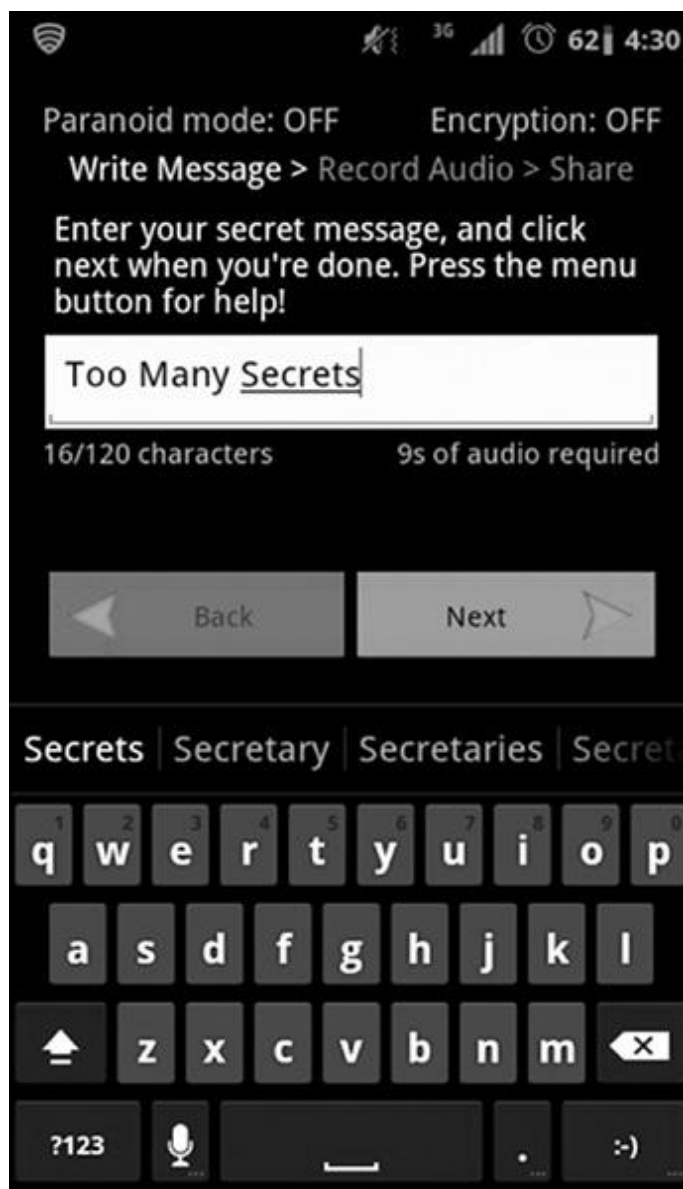
---

<sup>۱</sup> echo steganography

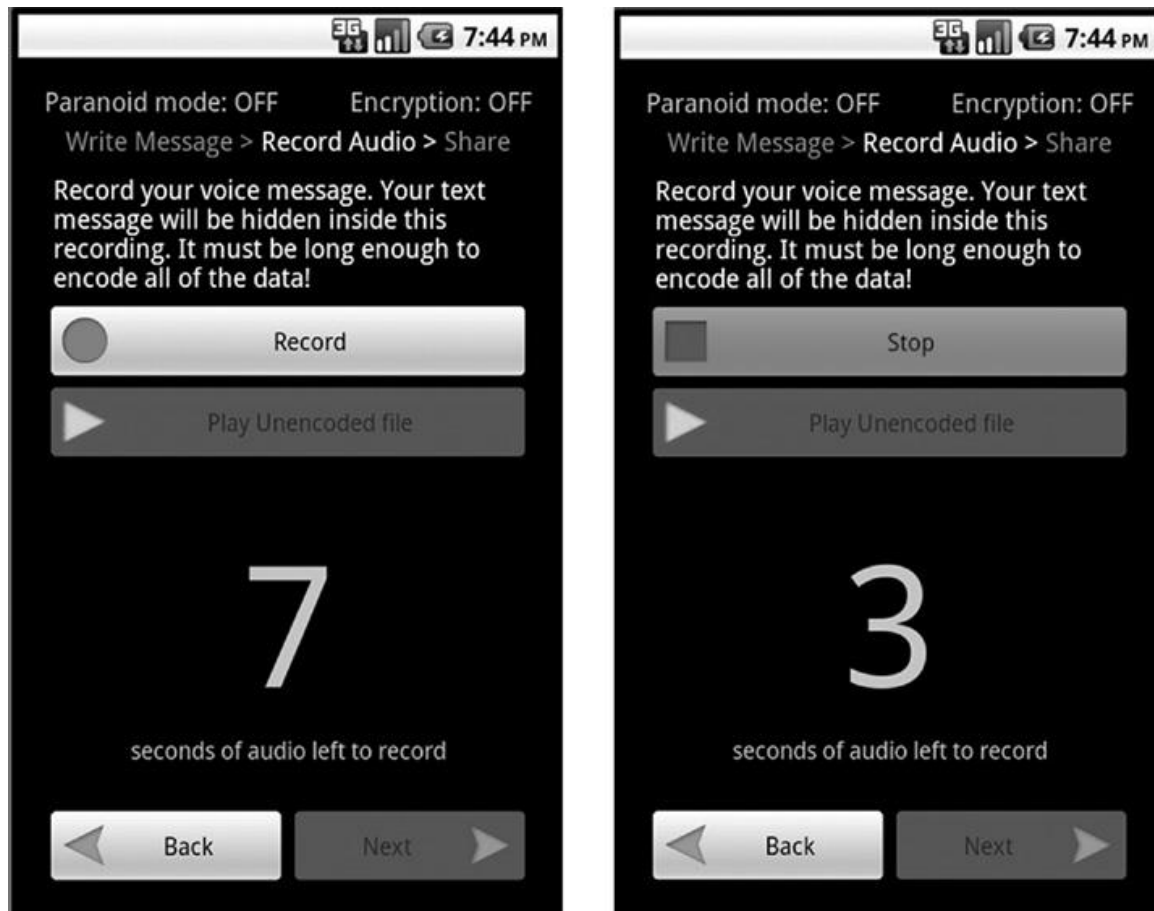
- (۲) در مقابل فشرده‌سازی با اتلاف در روش فشرده‌سازی Mp3 مقاوم است و روش قدرتری را برای پنهان‌سازی داده‌ها ایجاد می‌کند.
- (۳) نرخ بیتی جاسازی شده‌ی کمتری را در مقایسه با جایگزینی به شیوه‌ی تغییر کم‌ارزش‌ترین ارائه می‌دهد.
- (۴) در مقایسه با سایر روش‌های پنهان‌سازی داده‌ها، بهتر اجرا می‌شود. آزمون‌های تجربی نرخ بیتی ۱۶ بیت در ثانیه را نشان داده که در شرایط عادی هم دست‌یافتنی است.

### استفاده از نرم‌افزار STEGDROID

پس از اجرای نرم‌افزار، کاربر می‌بایست پیام سری را حداکثر در ۱۲۰ حرف تایپ کند. (عکس ۵-۲۲) پس از تایپ پیام، کاربر پیام صوتی را به وسیله‌ی میکروفون موجود در ابزار اندرویدی خود ضبط می‌کند. حداقل طول پیام ضبط‌شده بستگی به تعداد حروف پیام دارد. نرم‌افزار به محض رسیدن به طول مورد نیاز، کاربر را آگاه می‌کند (عکس ۵-۲۳).

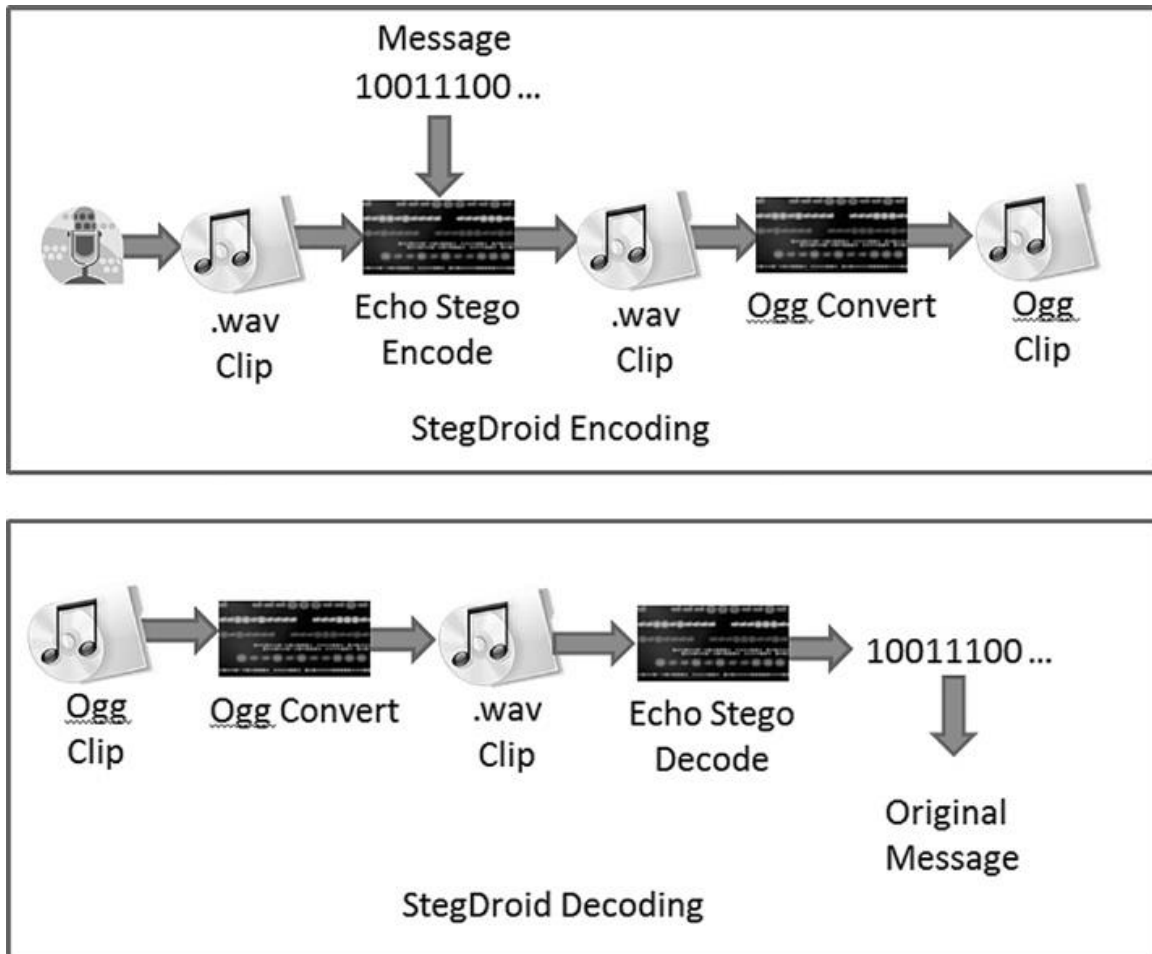


شکل ۵-۲۲: صفحه کلید نرم افزار STEGDROID



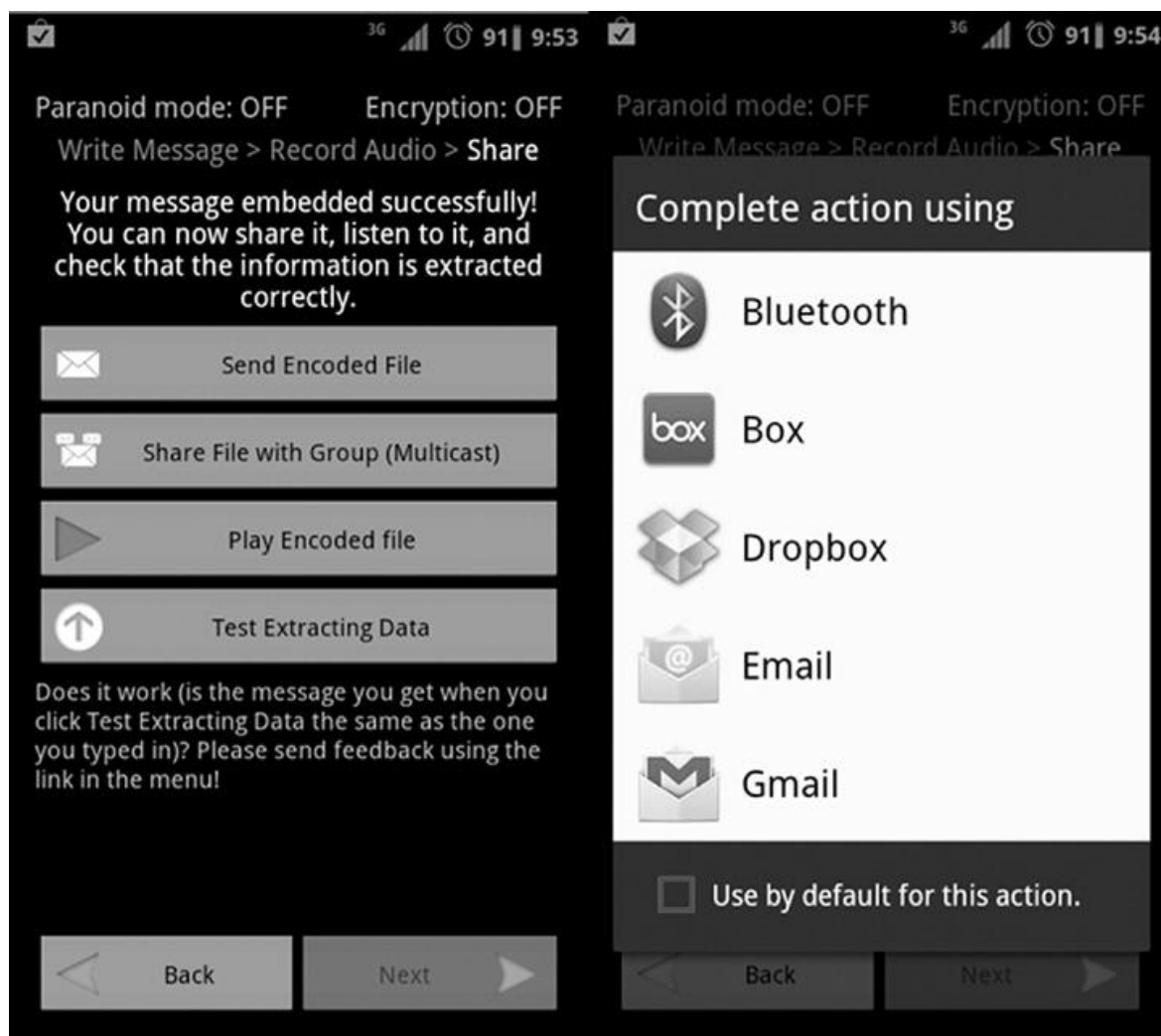
شکل ۵-۲۳: امکان ضبط فایل حامل در نرم‌افزار STEGDROID

در طول زمان پردازش به شیوه‌ی استتار انعکاسی، دو الگوریتم متفاوت انعکاسی به کار می‌رود، یک الگوریتم برای نگاشت یک باینری و دیگری نگاشت صفر باینری؛ در نتیجه با توجه به مقداری که می‌خواهیم پنهان کنیم (۰ یا ۱) الگوریتم مناسب انتخاب می‌شود. در زمان بازیابی داده‌های کد شده، الگوریتم تشخیص سعی در شناسایی انعکاس و تعیین انعکاس صفر یا انعکاس یک کرده و سپس مقادیر را بازیابی می‌کند و این کار تا پایان کلیپ صوتی ادامه می‌یابد. فایل صوتی حاصل که حاوی داده‌های پنهان شده است تحت فرمت منبع باز ogg ذخیره می‌شوند. فایل‌های ogg بیشتر برای ارسال از جریان کاراکترها در فایل‌های چندرسانه‌ای به کار می‌رود (عکس ۵-۲۴).



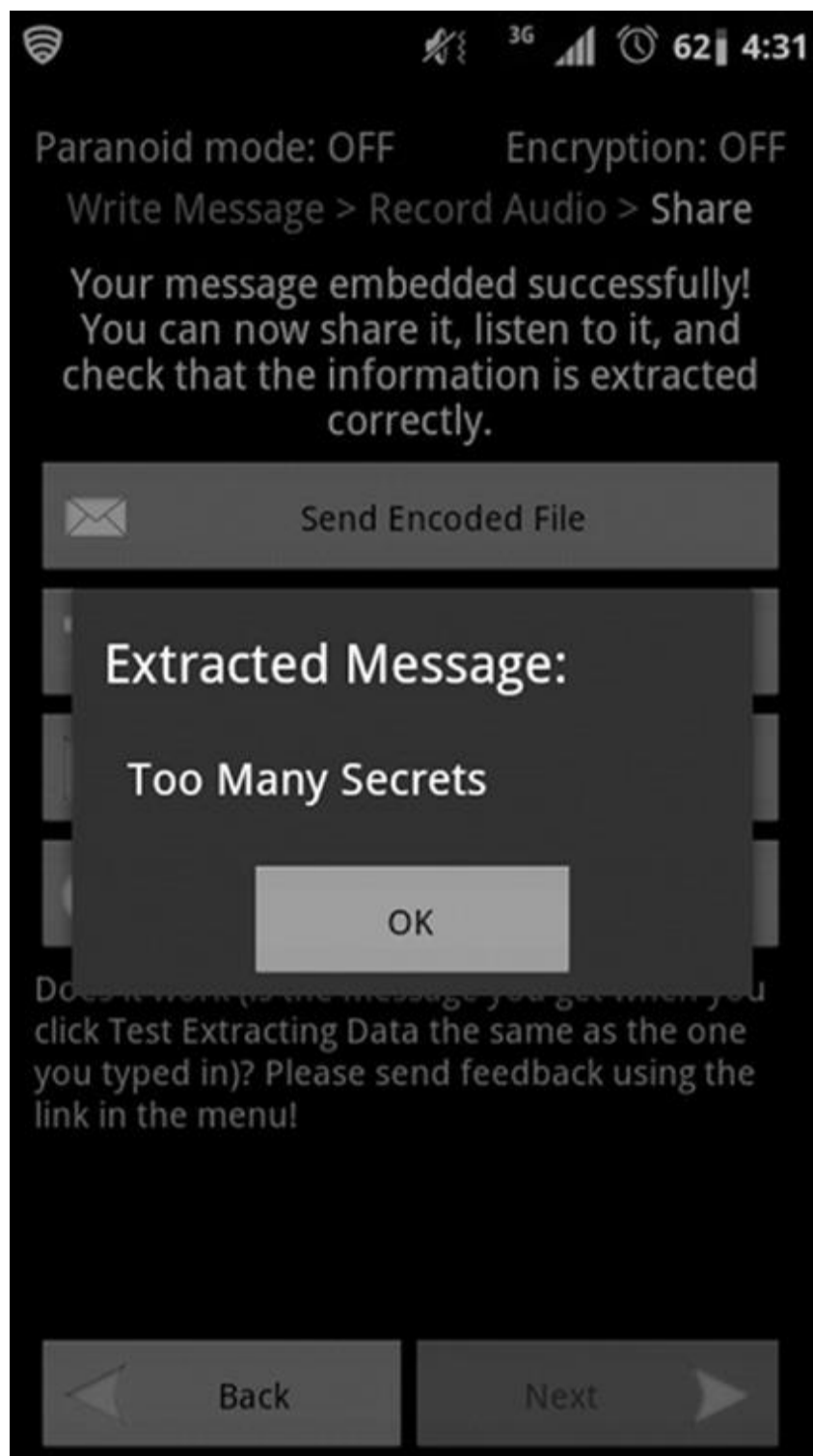
شکل ۵-۲۴: دیاگرام کد کردن و گدشایی کردن نرم افزار **STEGDROID**

پس از پایان جاسازی داده ها، نرم افزار امکان افزودن پیامی برای تأیید اعتبار فایل برای شنیدن پیام ضبط شده، یا ارسال آن به وسیله ی پست الکترونیکی یا سایر نرم افزارهای نصب شده دیگر را فراهم می کند. پس از ارسال فایل حاصل با پسوند **ogg** به کاربر دیگر، وی تنها نیاز به اجرای برنامه ی مشابه و بازیابی پیام پنهان شده است. شاید فکر کنید به راحتی با فیلتر کردن فایل هایی در قالب **ogg** می توان از انتقال پیام های پنهانی پیشگیری کرد ولی فایل های **ogg** برای بسیاری از مقاصد قانونی در ابزارهای اندرویدی مثل انتقال انواع زنگ موبایل استفاده می شود. توجه کنید که این نرم افزار روش هایی برای امنیت بیشتر مثل امکان رمزنگاری پیام، پس و پیش از جاسازی آن در فایل حامل و امکان پاک کردن باقیمانده های پیام در ابزار اندرویدی را دارد (عکس ۵-۲۵).



شکل ۵-۲۵: امکان ارسال فایل حامل پیام و مشاهده پیام در فایل دریافتی

پیام پنهان شده در فایل صوتی و بازیابی آن را در شکل ۵-۲۶ مشاهده نمایید.



شکل ۵-۲۶: STEGDROID با موفقیت پیام پنهان شده را بازیابی نماید

## چکیده

پنهان‌سازی داده‌ها در فایل‌های ویدیو غیرفشرده مثل AVI و فایل‌های فشرده مثل Mpeg امروزه نه تنها امکان‌پذیر است، بلکه ظرفیت چشمگیری برای داده‌های پنهان که به طور پیوسته می‌توان منتقل کرد را ارائه می‌دهد. با امکانات موجود و آتی تصحیح خطا و قابلیت افزایش داده‌ها، امکان قصر در رفتن داده‌های پنهان حتی در محیط‌های با نویز زیاد را می‌دهد و سطح دیگری از حملات را پیش روی ما قرار می‌دهد. پس تحلیل‌گران کار دشوار تشخیص یا دست‌کم، تنگ کردن<sup>۱</sup> کانال‌های پوششی ارسال آن‌ها را برای پیشگیری از ارتباطات در لفافه بر عهده دارد. وظیفه دیگر ناظران امنیتی پیشگیری از سرقت سرمایه‌های فکری سازمان‌ها یا استفاده از این کانال‌ها برای رساندن دستورات و کدهای کنترل بدافزاری که می‌تواند امکان حملات پیشرفته را آسان کرده یا شتاب دهد، می‌باشد.



## فصل ششم

### پنهان سازی داده در سیستم عامل اپل

پیدایش انفجارگونه‌ی نرم‌افزارهای نوین پنهان‌سازی در ابزارهای ios، به ویژه Ipad و iPhone بسیار چشمگیر است. این موضوع چه به خاطر نیاز به حفظ حریم شخصی هنگام استفاده از ابزارهای همراه باشد و چه در قالب نقطه شروع کار توسعه‌دهندگان نرم‌افزارهایی باشد که قصد دارند برای اهداف بدکارانه از آن استفاده کنند، دریچه‌ای برای پنهان‌سازی داده‌ها می‌باشد.

#### نرم‌افزارهای پنهان‌سازی داده‌ها در ابزارهای همراه

در کتاب مشهور کدشکنان<sup>۱</sup> نوشته‌ی David kahns به شخصی به نام Demaratus اشاره می‌کند که به سرزمین پارس اعزام شد و به نقشه‌ی حمله‌ی آن‌ها به یونان پی می‌برد. در ادامه داستان Demaratus می‌باید یک پیام سری را برای هشدار به Spartans بفرستد. ابزار نوشتن آن روزها لوح موم‌اندود بوده که به Ipad امروزی شبیه است، که تنها برد الکترونیکی و باتری لیتیوم آهن آن را کم داشت (شکل ۶-۱ و ۶-۲).

---

<sup>۱</sup> The code breakers



شکل ۶-۱: موم تبلت



شکل ۶-۲: ابزار نوشتن متن در iPad

Demaratus برای نوشتن پیام سری، موم را از لوح جدا کرده و با قلم نوک تیز اقدام به نوشتن بر روی چوب لوح نموده و دوباره لایه‌ای از موم را به عنوان پوشش بر روی آن قرار می‌دهد. این کار باعث شد این لوح، راهش را در میان نگهبانان پیدا کرده و به موقع برای دادن هشدار به یونانی‌ها به دستشان برسد. اگر لوح به دست Cleomenes می‌افتاد و او می‌دانست که می‌بایست موم را برداشته و پیام را کشف و بازیابی کند، چه آینده‌ای در انتظار اسپارته‌ها بود؟ با این کار Detn پیام را به اسپارته‌ها رساند و زمان کافی آماده شدن برای جنگ و تقویت استحکامات را به آن‌ها داد. جای تعجب دارد که دفعه‌ی بعد با استفاده از این روش اسپارته‌ها بتوانند پیام تاخت‌وتاز حتمی بعدی را به موقع دریافت کنند (عکس ۳-۶ و ۴-۶).



شکل ۶-۳: فایل استاندارد که به عنوان فایل حامل از آن استفاده می‌شود



شکل ۶-۴: فایل پنهانی که به عکس حامل اضافه می کنیم

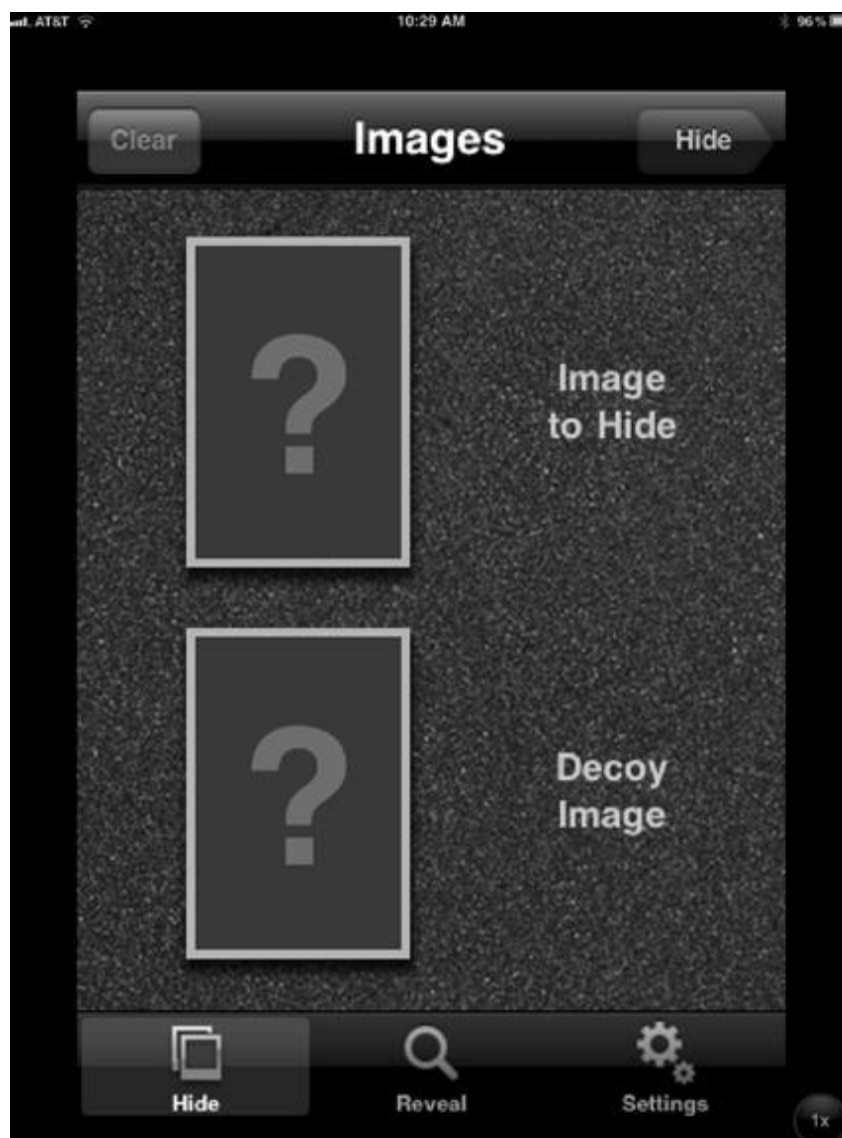
برای بررسی استتار در این فصل از Ipad و نرم افزار پنهان سازی داده همراه آن استفاده می کنیم. این نرم افزارها را می توان مستقیم و بدون هیچ گونه تغییری از سایت Itune دانلود کنیم. بسیاری از نرم افزارهای پنهان سازی داده ها در iPhone و Ipad و ابزارهای مشابه وجود دارد و ما برای بررسی و آزمودن روش و ویژگی های آنها، چند نرم افزار را به اسامی زیر انتخاب کرده ایم که ویژگی منحصر به فرد روش های پنهان سازی داده در هر کدام را شرح دهیم.

- 1) Spy Pix
- 2) Invisi Letter
- 3) Stego Sec

## تحلیل نرم افزار SPY



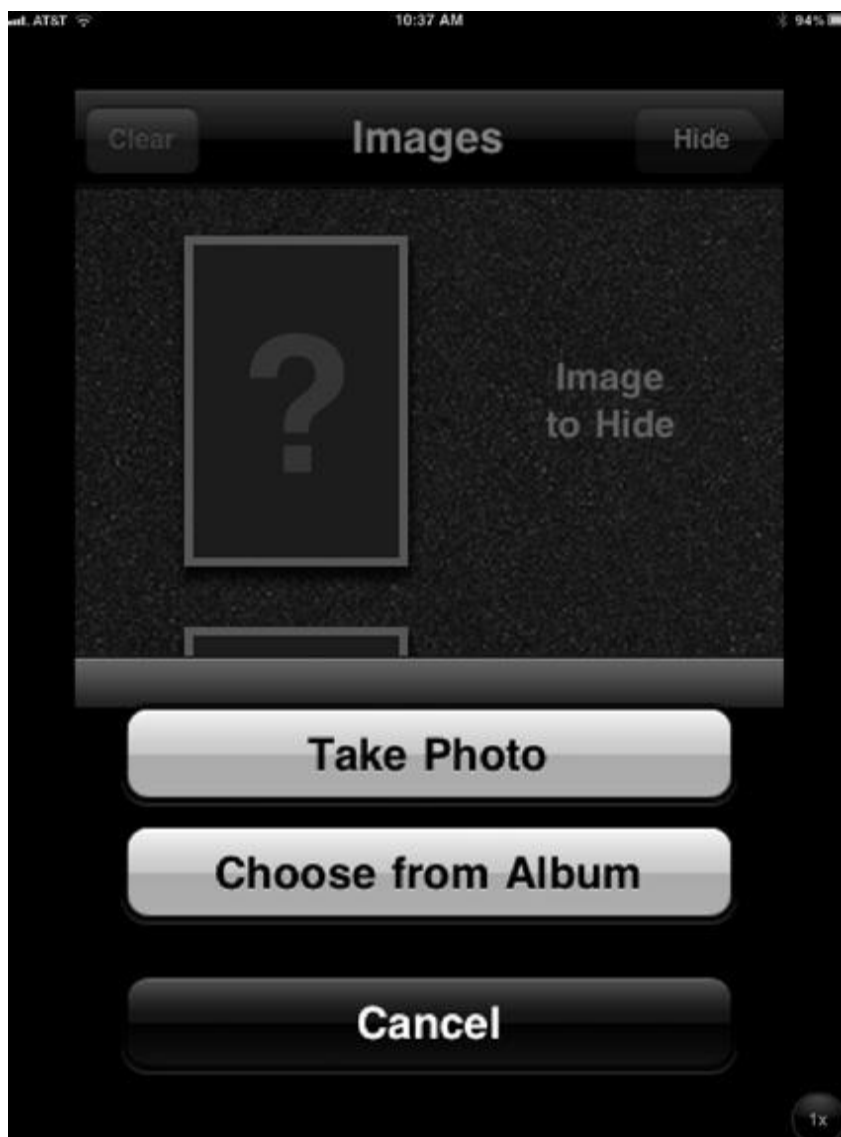
مشابه بسیاری از نرم افزارهای دیگر Ipad، نرم افزار Spy Pix، هم کاربردی آسان برای پنهان سازی داده دارد. این APP از روشی استفاده می کند که به کاربران اجازه می دهد پنهان سازی عکس را داخل عکس دیگر با کیفیت های گوناگون می دهد. تبعات این پنهان سازی، کاهش اندازه و کیفیت هر دو عکس سری و حامل است، ولی کشف آن همچنان مشکل باقی می ماند. صفحه ای که نرم افزار از آن شروع می شود را در شکل ۶-۵ مشاهده می کنیم.



شکل ۶-۵: پنجره انتخاب عکس در نرم‌افزار Spy Pix

| Spy Pix Details  |                |
|------------------|----------------|
| Application Name | Spy Pix        |
| Seller           | Juicy bits     |
| Image format     | True color PNG |
| Last release     | December 2009  |

شما باید دو عکس را مشخص کنید؛ عکس نخست عکسی است که می‌خواهید پنهان کنید (به عبارت دیگر عکس سری) و عکس دوم عکس پوششی است. با انتخاب عکسی سری پنجره شکل ۶-۶ نمایان می‌شود.



شکل ۶-۶: گزینه‌های انتخاب عکس حامل

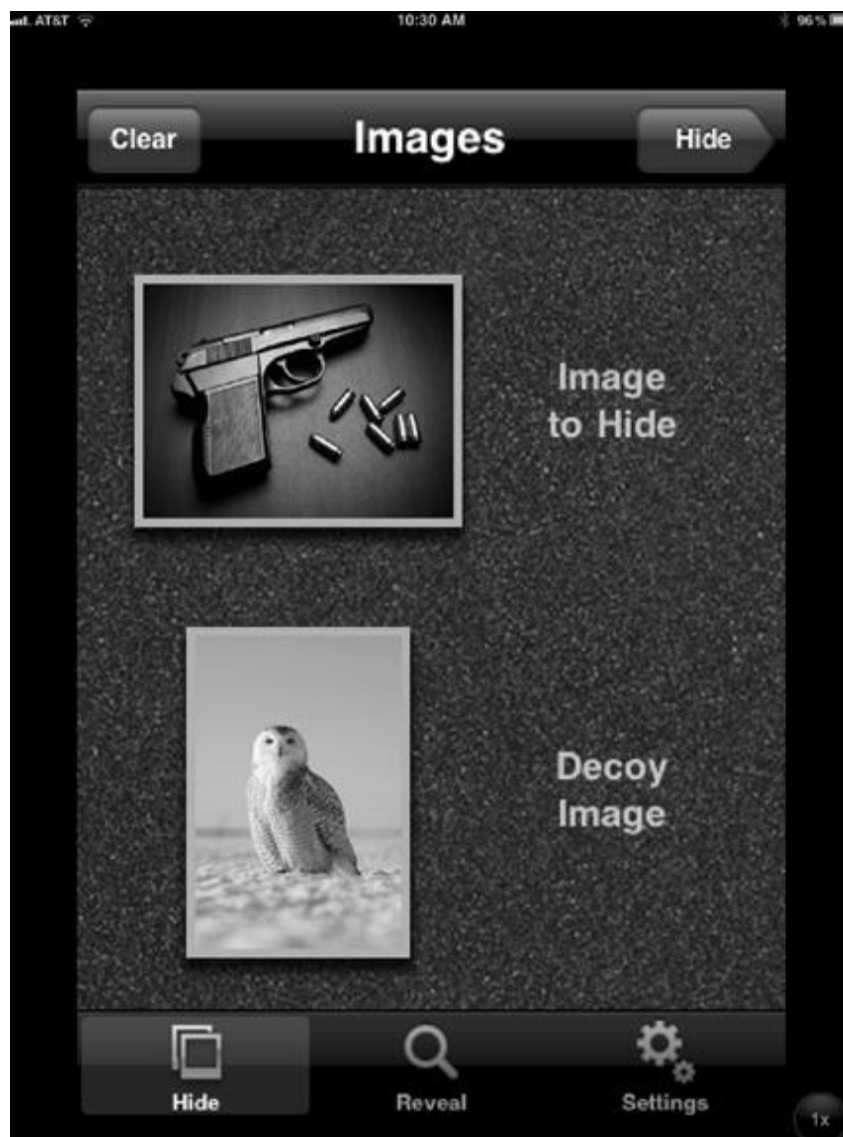
این نرم‌افزار به کاربر اجازه می‌دهد یا از عکس‌های موجود در آلبوم عکس استفاده کرده یا در همان زمان اقدام به گرفتن عکس کند. ما در این مثال از همان عکس اسلحه و فشنگ‌ها استفاده می‌کنیم (عکس ۶-۷).



شکل ۶-۷: عکس سری انتخاب شده در نرم‌افزار Spy Pix

همین مراحل هم برای عکس حامل به همان منوال تکرار می‌شود (عکس ۶-۸).





شکل ۶-۸: عکس سری و عکس حامل در نرم‌افزار Spy Pix

اکنون می‌توانیم دو عکس را با هم ترکیب کنیم. Slider پایین صفحه به کاربر اجازه می‌دهد که سطح پنهان‌سازی مورد نظرش را تعیین کند (عکس ۶-۹). با تغییر این سطح از بالا به پایین می‌توانید سطح مطلوب مورد نظر خود را انتخاب کنید. اگر این سطح خیلی پایین انتخاب شود عکس تفنگ و فشنگ‌ها در داخل عکس پوشش به وضوح نمایان است، اما اگر سطح مطلوب بالاتری را انتخاب کنید - مثل عکس ۶-۱۰ - عکس پوششی کاملاً عکس سری را ناپیدا و غیر قابل مشاهده می‌کند.



شکل ۶-۹: پنهان‌سازی با کیفیت کم



شکل ۶-۱۰: پنهان سازی عکس سری در عکس حامل

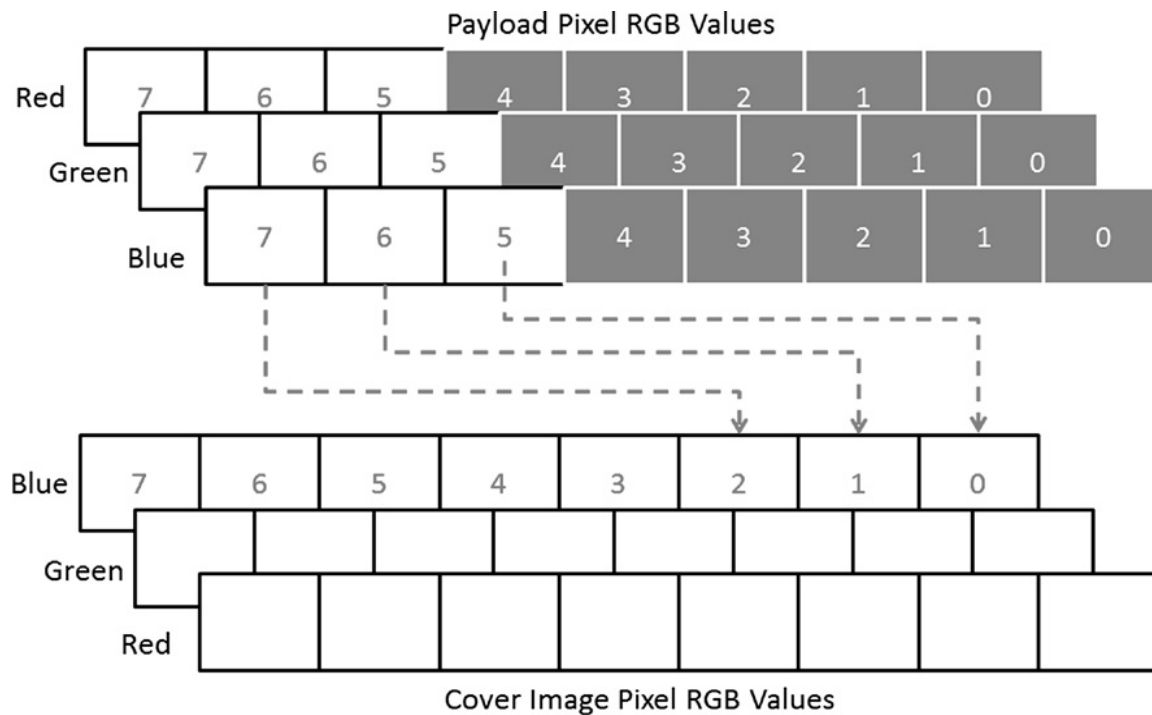
روش به کار رفته در Spy Pix بسیار ساده است. همپوشانی دو عکس به وسیله‌ی جایگزینی با ارزش‌ترین بیت‌های عکس سری با کم ارزش‌ترین بیت‌های عکس پوشش انجام می‌شود. بسته به کیفیت عکس سری، کیفیت عکس حامل هم کاهش می‌یابد.

### تحلیل روش پنهان سازی در نرم افزار SPY

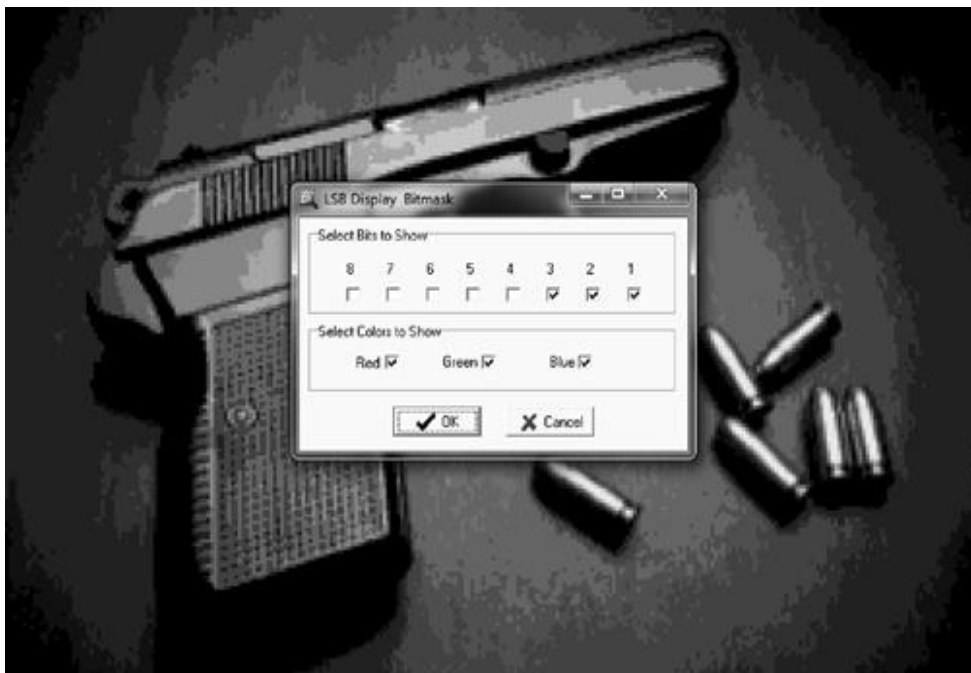
Spy برای یکسان کردن فرمت، هر دو عکس را به فرمت 24 bit True Color تبدیل می‌کند. این نرم افزار به کاربر اجازه می‌دهد که تعداد پیکسل‌هایی که در عکس پوشش باید تغییر کند را مشخص

نماید که از ۰ تا ۷ بیت است. اگر صفر را انتخاب کنید، تمام عکس پوشش جایگزین عکس سری شده و تصویر اصلی را خراب می‌کند. اگر ۷ را انتخاب کنید تنها مقادیر باارزش‌ترین بیت‌های عکس سری با کم-ارزش‌ترین بیت‌های عکس پوششی جایگزین می‌شود.

در عکس ۶-۱۱ سطح را ۵ انتخاب کرده‌ایم که باعث می‌شود مقدار بیت‌های ۵ و ۶ و ۷ هر بایت RGB عکس سری با بایت‌های ۰، ۱، ۲ از عکس پوشش جایگزین می‌شوند. همان‌گونه که در عکس زیر مشاهده می‌کنید، بیت‌های ۰ تا ۴ عکس سری نادیده گرفته می‌شوند که باعث می‌شود کیفیت عکس سری از ۲۴ بیت به ۹ بیت کاهش پیدا کند. هر رنگ قرمز، سبز و آبی در ۸ بیت، که جمعاً ۲۴ بیت نمایش داده می‌شود و ۵ بیت در هر رنگ حذف شده که جمعاً  $5 \times 3 = 15$  بیت می‌شود؛ بنابراین کیفیت عکس  $24 - 15 = 9$  می‌شود.



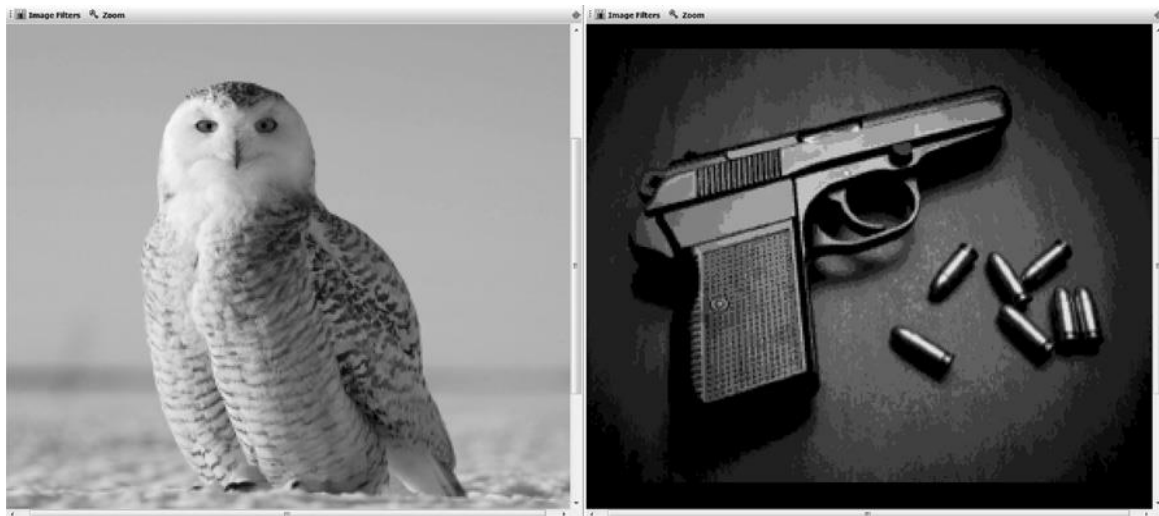
شکل ۶-۱۱: دیاگرام پنهان‌سازی داده‌ها در Spy Pix



شکل ۶-۱۲: پنجره انتخاب بیت های مورد نظر در پنهان سازی عکس

در نگاه اول شاید فکر کنید به سادگی از روی تصویر عکس حامل بتوان به صورت دیداری وجود عکس دیگر را تشخیص داد، اما عکس حامل کاملاً بی عیب به نظر می رسد، حتی اگر ۲ یا ۳ بیت را هم جایگزین می کردید باز همین طور بود. برای آشکار کردن عکس پنهان شده باید با عکس حامل به گونه ای دیگر رفتار کنیم. یک روش پردازش زدن با در نظر گرفتن تنها بیت های کم ارزش است.

برای تحلیل دوباره از نرم افزار Wetstone استفاده می کنیم و با مشخص کردن دقیق تعداد بیت های کم ارزش ترین بیت عکس حامل و رنگ های آن می توان پردازش زدن عکس را کنترل کرد. در عکس ۶-۱۳ دو عکس را در کنار هم مشاهده می کنیم. عکس سمت چپ عکس حاوی عکس سری اسلحه و فشنگ در نمایش عادی است. عکس سمت راست همان عکس با پردازش زدن بیت های مشخص شده و رنگ های تعیین شده در شکل ۶-۱۲ است. با مشاهده ی دو عکس در کنار هم، کاهش وضوح تصویر جغد برفی (فایل حامل) که تصویر سری در آن پنهان شده نمایان است.



شکل ۶-۱۳: مشاهده عکس سری (سمت راست) و عکس حامل که عکس سری در آن پنهان شده (سمت چپ) در کنار هم

از آن جایی که تنها ۳ بیت پرارزش عکس سری در عکس حامل ذخیره می‌شود از دست رفتن داده‌های فایل عکس سری چشمگیر است؛ باز اطمینان داریم که عکس تفنگ و فشنگ‌ها قابل تشخیص هستند.

شاید با باقی ماندن ۹ بیت از ۲۴ بیت داده‌های عکس اصلی تصور کنید که تشخیص الگوریتم به راحتی امکان‌پذیر است؛ ولی این‌گونه نبوده و با استفاده از روش کلی الگوریتم شناسایی و تشخیص تغییر کم ارزش‌ترین بیت و تحلیل آماری مقادیر آن می‌توان به وجود داده‌های نهان پی برد. بسیاری از نرم-افزارها، نخست اطلاعاتی که می‌خواهند پنهان کنند را فشرده کرده، سپس نسبت به تغییر کم ارزش‌ترین بیت فایل حامل با این داده‌ها اقدام می‌کنند و این کار باعث می‌شود تغییر کم ارزش‌ترین بیت فایل حامل بختانه به نظر برسد. در مورد مثال پیش، داده‌های پنهان شده در فایل حامل، شکل تصادفی بسیار کمی دارند، زیرا مقادیر پر ارزش‌ترین بیت از عکس سری بسیار کمتر از مقادیر کم ارزش‌ترین همان عکس تغییر می‌کنند و این تغییرات حتی از تغییرات پس از فشرده‌سازی داده‌ها هم کمتر است. در تحلیل، برای برآورد امکان این نوع تشخیص، نیازمند مدل‌های مقایسه‌ای جدید و رویکردهای آموزش در شبکه‌های عصبی که برای تشخیص تغییراتی غیر نرمال که توسط روش‌های پنهان‌سازی داده به کار می‌رود می-باشیم.

رویکرد پایه برای توسعه‌ی چنین روش‌های شناسایی، ایجاد گرد آیه‌ای بزرگ از نمونه‌هایی است که از این شیوه استفاده کرده‌اند و توسعه‌ی روش‌های اندازه‌گیری آماری با مقایسه این مجموعه گردآوری شده با عکس اصلی و تعیین تفاوت‌های کم ارزش‌ترین بیت این مجموعه که محتوای مقادیر کم ارزش‌ترین

بیت جایگزین شده با طول متغیر هستند. مراحل آموزش شبکه عصبی یا روش های اکتشافی دیگر، آن قدر ادامه پیدا می کند تا به بیشترین سطح دقت رسیده و تشخیص اشتباه را کاهش دهد.

## تحلیل stego sec



| Stego Sec Details |                     |
|-------------------|---------------------|
| Application Name  | Stego Sec           |
| Seller            | Raffaele De Lorenzo |
| Image format      | JPEG                |
| Last release      | February 2011       |

این App، امکان پنهان کردن متن در داخل عکسی که در همان لحظه گرفته شود یا از عکس های موجود در گالری iPhone و iPad انتخاب شده را فراهم می کند. عکس ۶-۱۴ مرورگر برنامه را نشان می دهد.

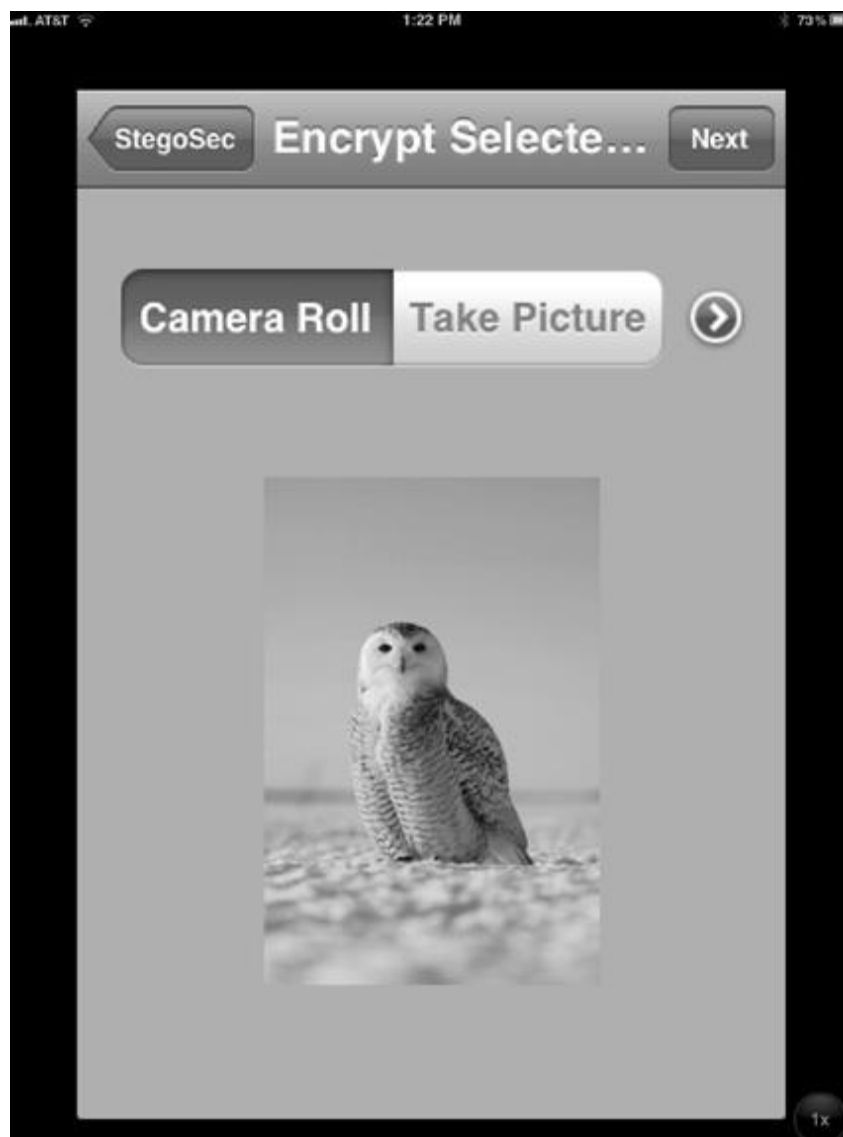


شکل ۶-۱۴: پنجره اصلی نرم‌افزار Stego sec

ولی ما بر روی چگونگی پردازش Crypto Image تمرکز می‌کنیم، به عبارت دیگر فقط چگونگی نهان‌سازی داده‌ها را بررسی می‌کنیم.

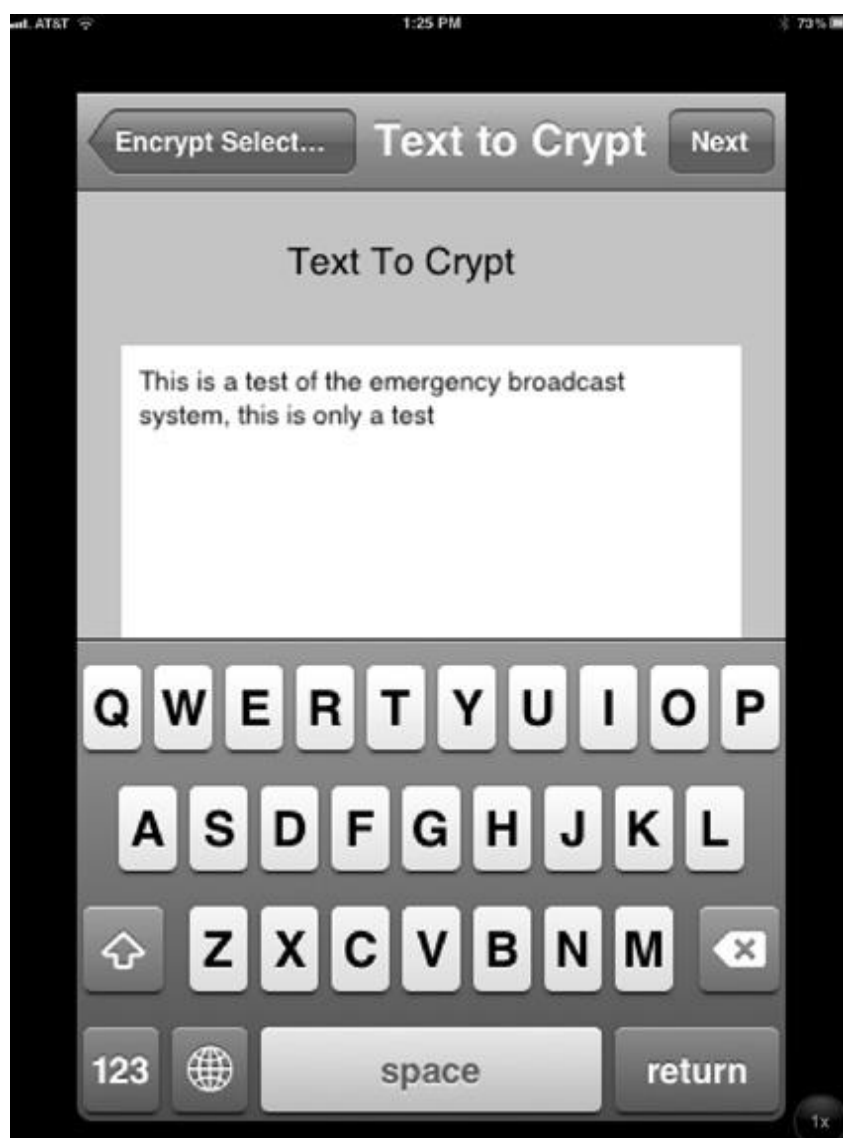
نرم‌افزار Stego sec مثل بسیاری دیگر از App موبایل، به تنهایی اجازه انتخاب بین عکس‌های موجود یا گرفتن عکس جدید در زمان پنهان‌سازی داده را می‌دهد. ما از همان عکس مثال‌های پیش برای پیوستگی مطالب با آن‌ها استفاده می‌کنیم (عکس ۶-۱۵).





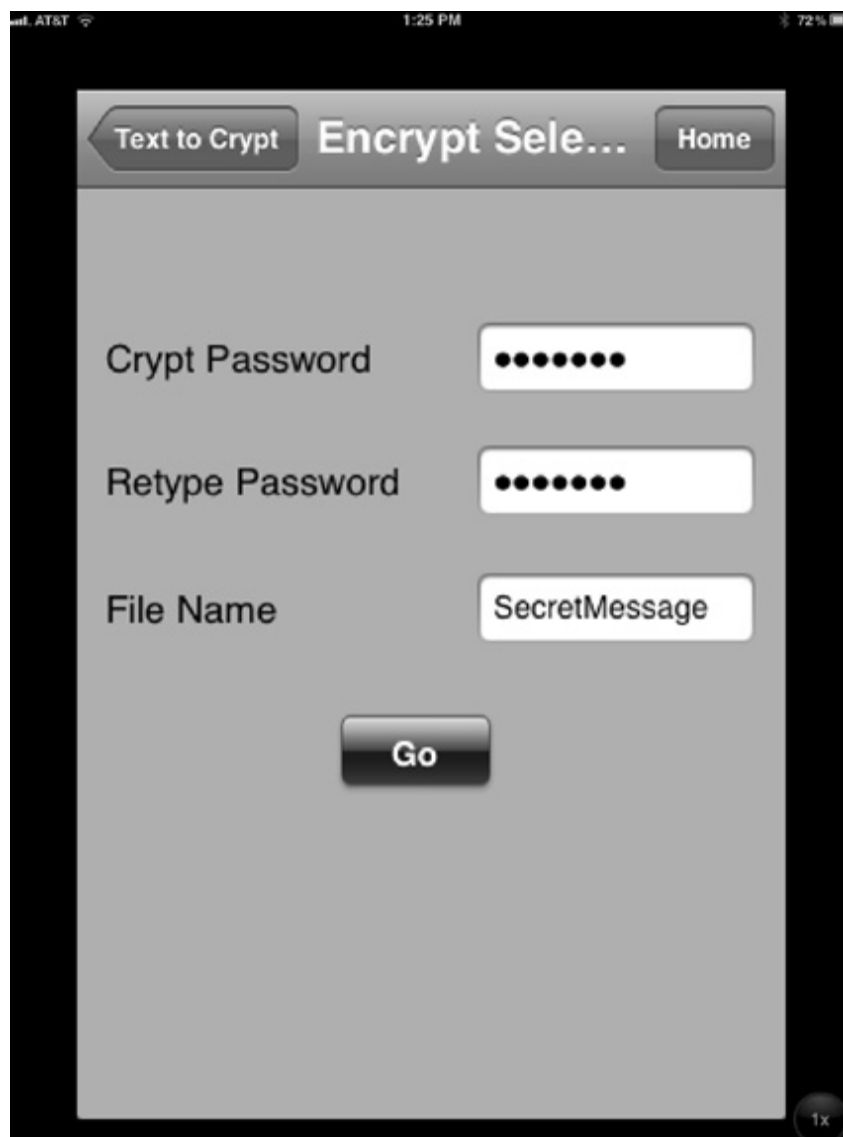
شکل ۶-۱۵: انتخاب فایل حامل از بین عکس‌های موجود یا گرفتن عکس جدید در همان لحظه

در مرحله بعد، باید پیامی که می‌خواهیم پنهان نماییم را تایپ کنیم. (عکس ۶-۱۶)



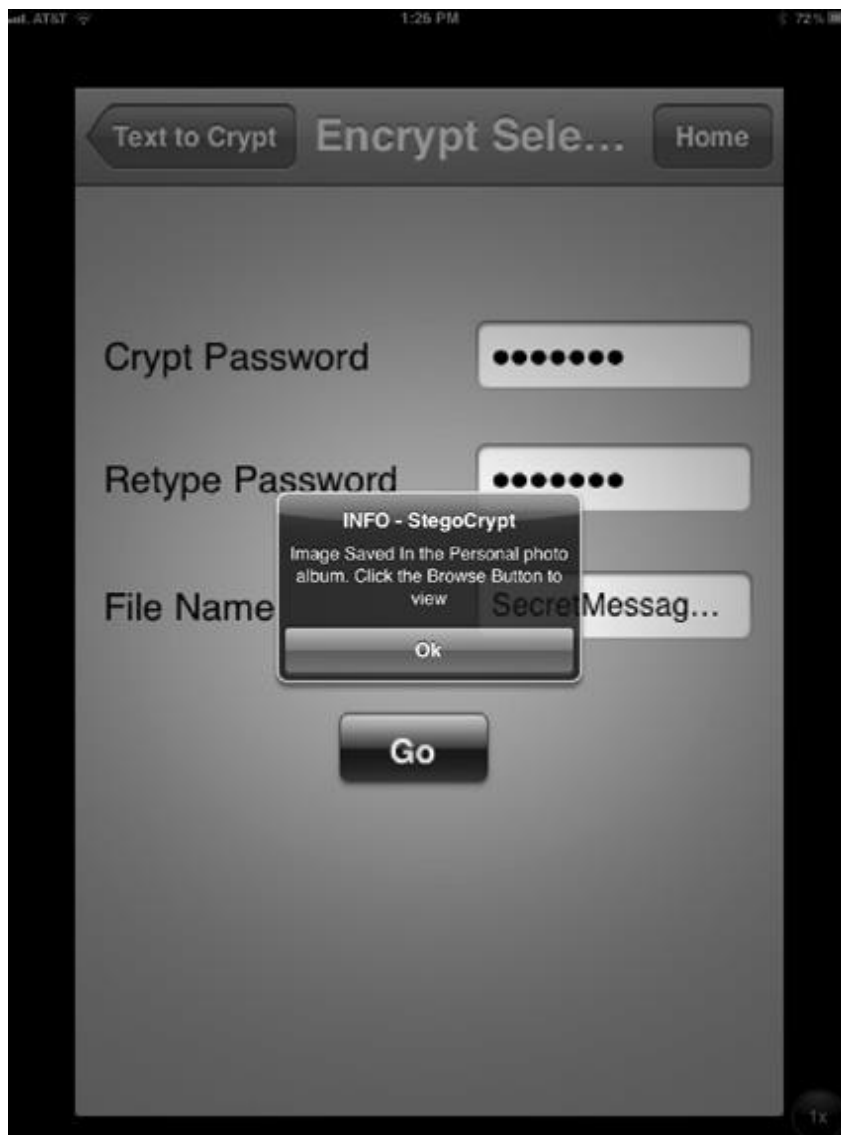
شکل ۶-۱۶: متن پیامی که می‌خواهیم در فایل عکس پنهان کنیم

نرم‌افزار Stego sec امکان رمزنگاری پیام، پیش از پنهان کردن آن را هم دارد. برای استفاده از این امکان باید گذر واژه تعیین کنید. در همین پنجره نام فایل را جهت ذخیره باید مشخص کنید. (عکس ۶-۱۷)



شکل ۶-۱۷: پنجره تعیین نام فایل حاصل و گذر واژه رمزنگاری آن

با انتخاب گزینه Go مراحل بالا تأیید شده و عکس سری ایجاد می شود (عکس ۶-۱۸).

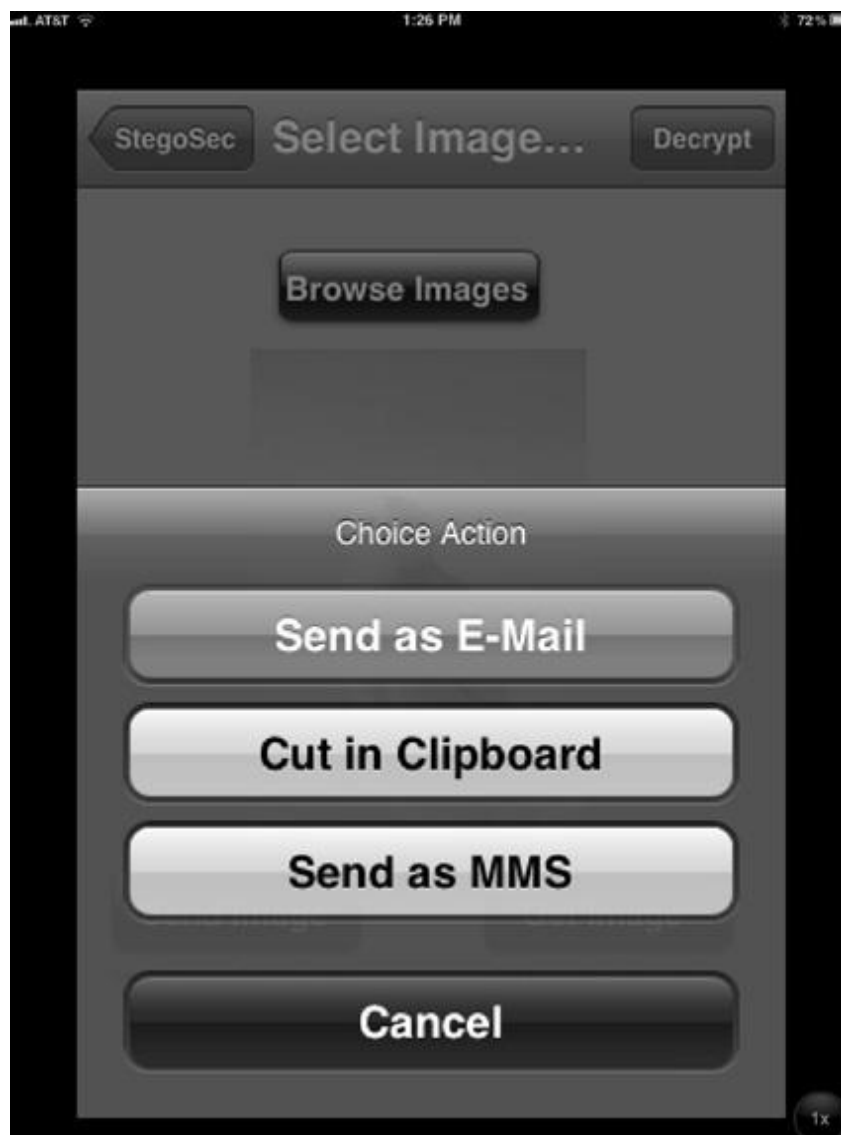


شکل ۶-۱۸: پایان موفقیت آمیز ایجاد فایل حامل

در پایان این مرحله می توانیم یا پیام پنهان را آشکار کرده یا مهم تر از آن پیام را به گیرنده ی مورد نظر ارسال نماییم. Stego sec قابلیت ارسال فایل حاوی پیام پنهان را به صورت پست الکترونیک یا MMS دارد (عکس ۶-۱۹ و ۶-۲۰).



شکل ۶-۱۹: نمایش یا ارسال پیام سری داخل فایل عکس



شکل ۶-۲۰: راه‌های گوناگون ارسال فایل

## روش‌های تحلیل پنهان‌سازی داده‌ها

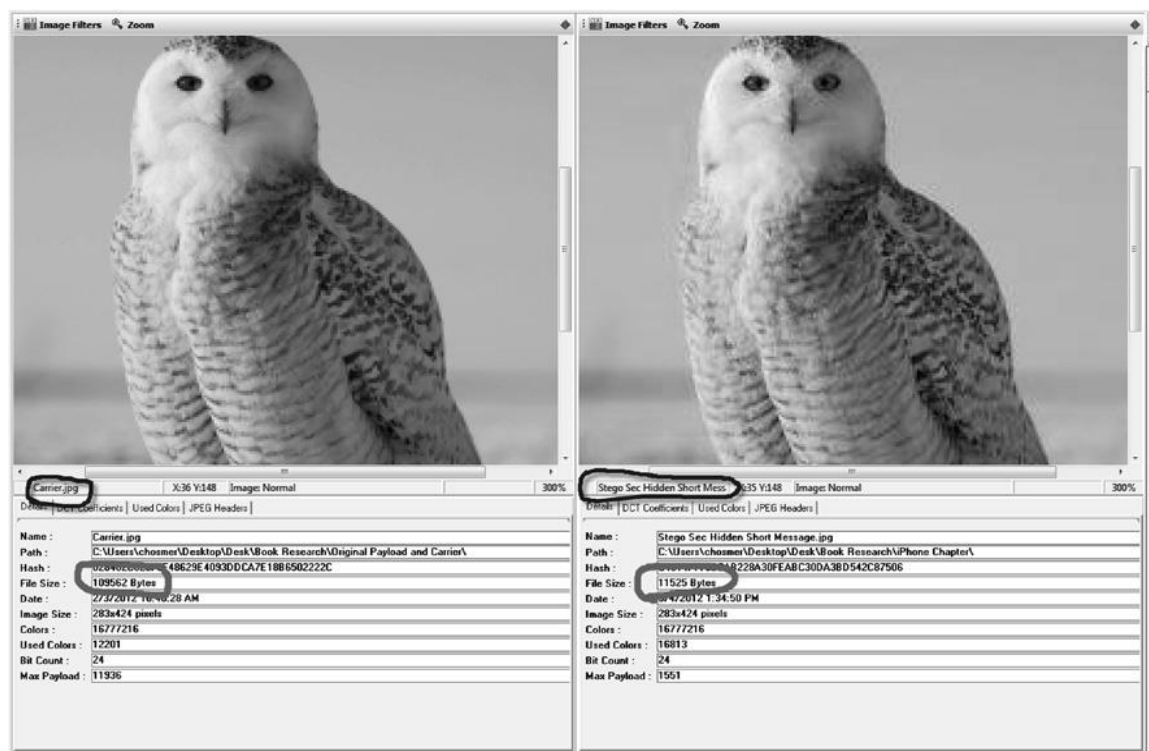
همان‌گونه که پیش‌تر اشاره شد، نرم‌افزار Stego sec پنهان‌سازی داده‌های را در فایل‌هایی از نوع Jpeg انجام می‌دهد. اکنون تمام قطعات جورچین برای تحلیل چگونگی پنهان‌سازی داده‌ها را به شکل زیر در اختیار داریم:

- (۱) فایل تصویری Jpeg اصلی
- (۲) فایل تصویری از نوع Jpeg حاوی پیام پنهانی
- (۳) پیام به طول ۷۰ حرف

(۴) گذر واژه ای که برای رمزنگاری داده به کار بردیم

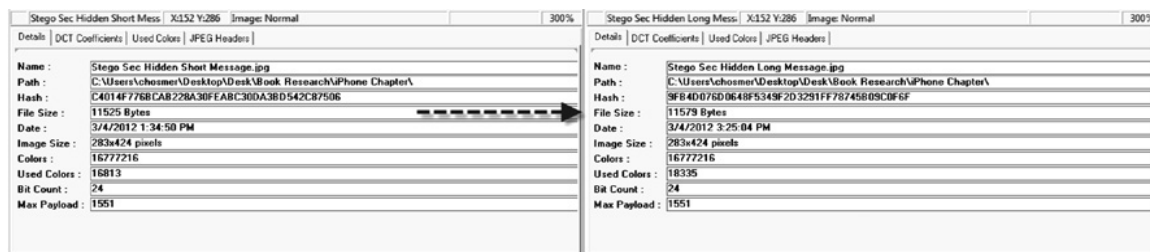
حال می توانیم تغییرات فایل را پیش و پس از پنهان سازی داده ها بررسی کرده و سعی می کنیم که دست کم روشی که برای پنهان سازی اطلاعات به کار رفته را پیدا کنیم.

برای تحلیل نتایج بررسی عکس و کشف روش به کار رفته در پنهان سازی، دوباره از نرم افزار به Stego Analyst کمک می گیریم. در عکس ۶-۲۱ تصویر سمت چپ عکس اصلی و تغییر نکرده جغد برفی را مشاهده می کنید. عکس سمت راست توسط Stego sec ایجاد شده و شامل پیام پنهان شده است که باید ارسال شود. در عکس ۶-۲۲ تفاوتی قابل توجه در اندازه ی فایل پیش و پس از افزودن پیام پنهان را مشاهده می کنید.



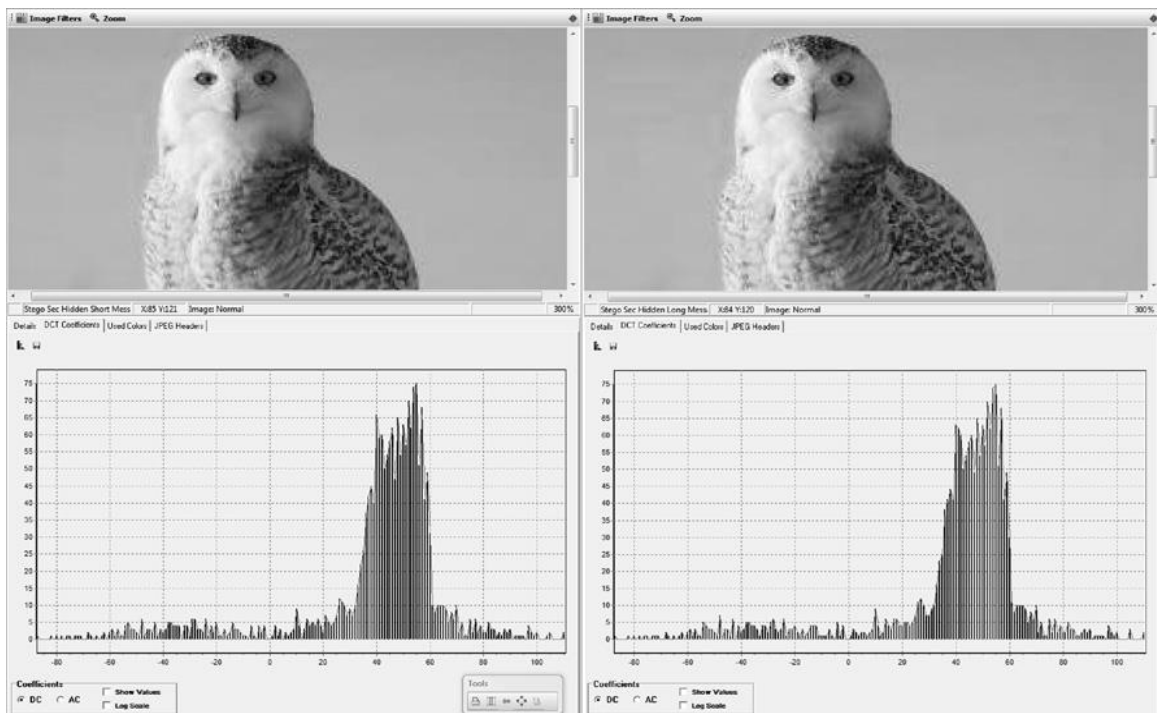
شکل ۶-۲۱: مشاهده همزمان فایل اصلی و فایل با داده های پنهان و مشاهده تغییرات حاصل

فایل اصلی ۱۰۹۵۶۲ بایت و پیام پنهان شده ۱۱۵۲۵ بایت است.



شکل ۶-۲۲: نرم‌افزار Stego Analyst جزئیات دو عکس با متن پنهان کوتاه و نهان‌شده در فایل را بررسی می‌نماید.

آنی می‌توانیم نتیجه بگیریم که روش به کار رفته در پنهان‌سازی داده‌ها، کدکردن دوباره‌ی فایل Jpeg با تغییر بردار DCT است، زیرا نیازی به کدکردن دوباره‌ی محتویات عکس نبوده است. به نظر می‌رسد که هیچ فیلد توضیح یا داده اضافه شده به فایل Jpeg وجود ندارد. همچنین مشخص شد که هیچ ساختار غیرعادی هم وجود ندارد. اکنون می‌توانیم آزمون دیگری برای اثبات فرض پنهان‌سازی داده‌ها با تغییر مقادیر DCT انجام دهیم. برای کمک به بررسی و مقایسه‌ی نتیجه، متن دیگر را با استفاده از Stego sec در فایل جاسازی کردیم که به جای ۷۰ حرف ۳۵۰ حرف دارد و از گذر واژه مشابه استفاده کردیم و این کار را به این خاطر انجام دادیم که فقط یک بعد (طول پیام) را تغییر دهیم. در شکل ۶-۲۳ دو عکس را با هم مقایسه می‌کنیم.



شکل ۶-۲۳: مقایسه ضرایب بردار DCT در پیام با طول کوتاه و بلند نهان شده در فایل پوشش یکسان

همان‌گونه که مشاهده می‌کنید تفاوت حاصل از پیام پنهان کوتاه و بلند در فایل Jpeg اندک است.



|                           |          |
|---------------------------|----------|
| Long Message Image Size:  | 11,579   |
| Short Message Image Size: | 11,525   |
| A difference of only      | 54 bytes |

با این وجود تفاوت اندازه‌ی پایه‌بار برابر  $۲۸۰ = ۷۰ - ۳۵۰$  بیت است و این موضوع فرض ما را که پیام متنی نهانی به روش ساده‌ی افزودن به بخش گیردیداری عکس اضافه شده است را تأیید می‌کند. برای بررسی دقیق‌تر تفاوت ایجاد شده به وسیله‌ی پیام کوتاه و پیام بلند باید مقادیر DCT هر دو عکس را در کنار هم به طور مستقیم آزمایش کنیم. همان‌گونه که در شکل ۶-۲۳ مشاهده می‌کنید هیستوگرام ضرایب DCT هر دو عکس شبیه هم به نظر می‌رسند. این نمودار به سادگی تعداد تکرار مقادیر در جدول مربوطه DC را در عکس نمایش می‌دهد. شکل ۶-۲۴ ضرایب DCT را که ما به عنوان مقادیر ضرایب DC به آن رجوع می‌کنیم را نمایش می‌دهد.

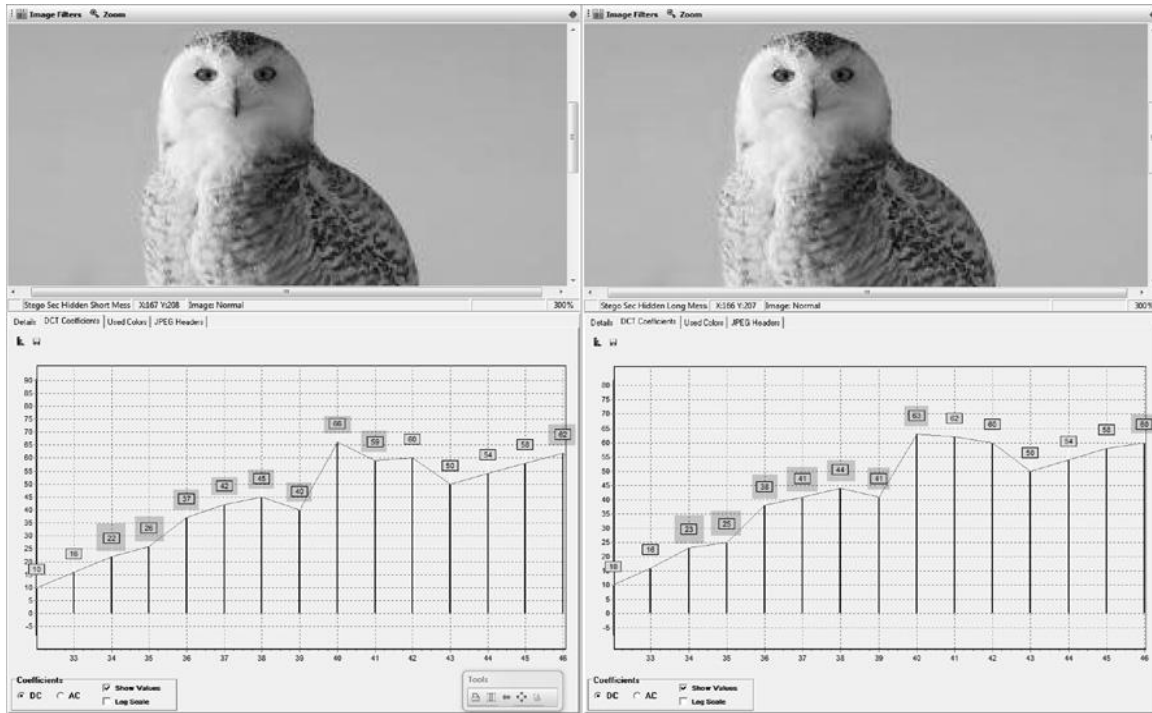
Quantized DCT Table Format

|   | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  |
|---|----|----|----|----|----|----|----|----|
| 1 | 0  | 1  | 5  | 6  | 14 | 15 | 27 | 28 |
| 2 | 2  | 4  | 7  | 13 | 16 | 26 | 29 | 42 |
| 3 | 3  | 8  | 12 | 17 | 25 | 30 | 41 | 43 |
| 4 | 9  | 11 | 18 | 24 | 31 | 40 | 44 | 53 |
| 5 | 10 | 19 | 23 | 32 | 39 | 45 | 52 | 54 |
| 6 | 20 | 22 | 33 | 38 | 46 | 51 | 55 | 60 |
| 7 | 21 | 34 | 37 | 47 | 50 | 56 | 59 | 61 |
| 8 | 35 | 36 | 48 | 49 | 57 | 58 | 62 | 63 |

شکل ۶-۲۴: مقادیر بردار DCT برای مقادیر میانگین

برای مشخص کردن تفاوت بین مقادیر گوانیتزه، نگاه دقیق‌تری به مقادیر گسسته‌ی مجموعه‌ی هیستوگرام می‌اندازیم تا نتایج آزمایش اختلافات جزئی در هیستوگرام ضرایب DCT را آشکار کنیم. همان‌گونه که در شکل ۶-۲۵ مشاهده می‌کنید، مقادیر مشخص شده، تغییرات جزئی در تفاوت اندازه‌ی پیام کوتاه و پیام بلند را نشان می‌دهند. از آنجایی که با تصاویر یکسانی شروع کردیم و گذر واژه مشترک هم به کار بردیم و تنها تفاوت در طول پیام نهان شده بود، پس نتیجه می‌گیریم که روش پنهان‌سازی بر پایه-

ی تغییر DC با مقادیر ضرایب DCT کار می‌کند. در نتیجه روش پنهان‌سازی Stego sec تغییر مستقیم مقادیر اتلاف‌پذیر<sup>۱</sup> تصاویر Jpeg است.



شکل ۶-۲۵: تغییرات هیستوگرام حاصل از پیام کوتاه و پیام بلند

بنابراین اندازه‌ی پیام نسبتاً کم و حدود چند صد بایت بود و چون توانایی تشخیص آماری آن در تحلیل روش به کار رفته در نرم‌افزار به وسیله‌ی انسان دشوار است، پس شیوه‌ی کار مورد استفاده در Stego sec به راه‌حل باارزش پنهان‌سازی داده‌ها (دست‌کم در پیام‌های متنی کوتاه) بدل شده است. نکته مهم دیگر بدون در نظر گرفتن نتایج تحلیل Stego sec زمانی که نسخه قبلی App را تحلیل نمودیم، این است که داده‌های پنهان در واقع در سرایند فایل ذخیره می‌شود که محل اولیه برای تشخیص و بازیابی آن است. شکل ۶-۲۶ نمایش مبنای ۱۶ عکس مورد استفاده در نسخه پیشین را نمایش می‌دهد. متن پنهان در این نسخه از نرم‌افزار در سرایند فایل جاسازی شده است.

<sup>۱</sup> lossy

|         | 0001 | 0203 | 0405 | 0607 | 0809 | 0A0B | 0C0D | 0E0F | 0123456789ABCDEF |
|---------|------|------|------|------|------|------|------|------|------------------|
| 0x00000 | ffd8 | ffe1 | 2ffe | 4578 | 6966 | 0000 | 4d4d | 002a | ÿØÿá/bExif..MM.* |
| 0x00010 | 0000 | 0008 | 0006 | 0112 | 0003 | 0000 | 0001 | 0001 | .....            |
| 0x00020 | 0000 | 011a | 0005 | 0000 | 0001 | 0000 | 0056 | 011b | .....v..         |
| 0x00030 | 0005 | 0000 | 0001 | 0000 | 005e | 0128 | 0003 | 0000 | .....^.(....     |
| 0x00040 | 0001 | 0002 | 0000 | 0213 | 0003 | 0000 | 0001 | 0001 | .....            |
| 0x00050 | 0000 | 8769 | 0004 | 0000 | 0001 | 0000 | 0066 | 0000 | ..+i.....f..     |
| 0x00060 | 00ec | 0000 | 0048 | 0000 | 0001 | 0000 | 0048 | 0000 | ..i...H.....H..  |
| 0x00070 | 0001 | 0008 | 9000 | 0007 | 0000 | 0004 | 3032 | 3231 | ....0221         |
| 0x00080 | 9101 | 0007 | 0000 | 0004 | 0102 | 0300 | 9286 | 0007 | .....'t..        |
| 0x00090 | 0000 | 0020 | 0000 | 00cc | a000 | 0007 | 0000 | 0004 | ... ..i .....    |
| 0x000a0 | 3031 | 3030 | a001 | 0003 | 0000 | 0001 | 0001 | 0000 | 0100 .....       |
| 0x000b0 | a002 | 0004 | 0000 | 0001 | 0000 | 0640 | a003 | 0004 | .....@ ...       |
| 0x000c0 | 0000 | 0001 | 0000 | 04b0 | a406 | 0003 | 0000 | 0001 | .....h..         |
| 0x000d0 | 0000 | 0000 | 0000 | 0000 | 4153 | 4349 | 4900 | 0000 | .....ASCII...    |
| 0x000e0 | 5468 | 6973 | 2069 | 7320 | 6120 | 6869 | 6464 | 656e | This is a hidden |
| 0x000f0 | 206d | 6573 | 7361 | 6765 | 0006 | 0103 | 0003 | 0000 | message.....     |
| 0x00100 | 0001 | 0006 | 0000 | 011a | 0005 | 0000 | 0001 | 0000 | .....            |

شکل ۶-۲۶: روش ابتدایی پنهان سازی داده ها در نسخه قبلی نرم افزار Stego sec

با نگاهی به آینده و پیشرفت های آن انتظار داریم که این App نه تنها کاربرد آسان داشته باشد، بلکه کیفیت روش های پنهان سازی به کار رفته را نیز گسترش دهد.

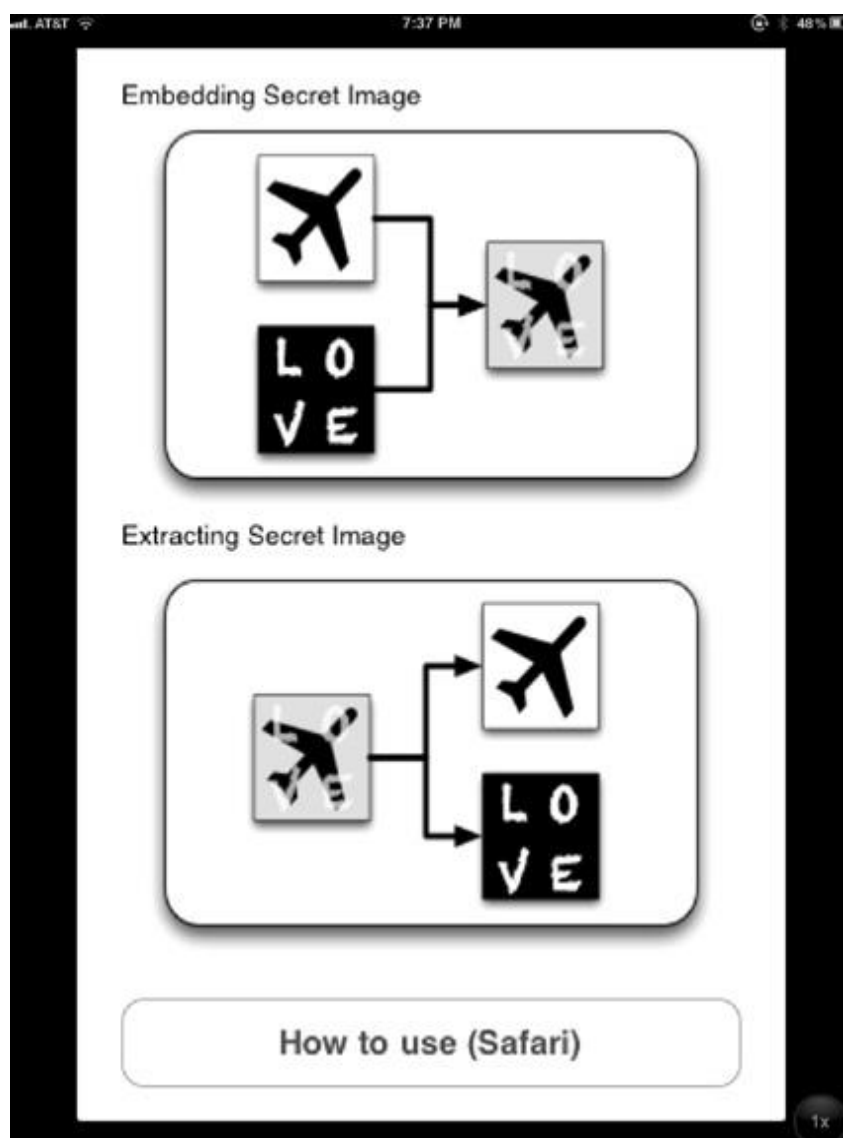
## تحلیل گره Invisiletter Anglysis



نرم افزار جالب دیگری برای ipad/iphone است و این App مثل سایر App تحت ipad/iphone عمل می کند ولی با کمی پیچیدگی بیشتر. همان گونه که در شکل ۶-۲۷ مشاهده می کنید، زمانی که نرم افزار را اجرا می کنیم امکان آشکار کردن پیام پنهان پیشین یا جاسازی پیام سری در یک عکس جدید را داریم.

البته توجه ما بیشتر به چگونگی پنهان سازی داده هاست. پس گزینه Embedding Secret

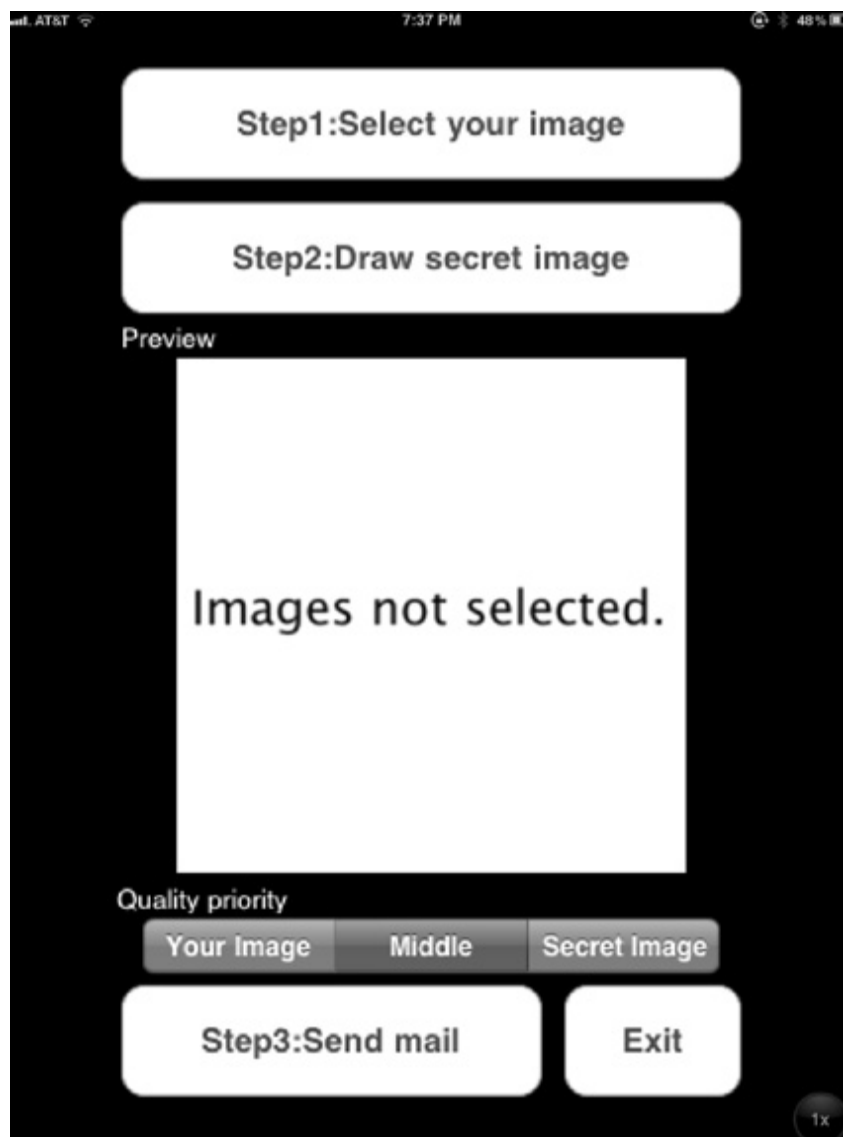
Image را انتخاب کرده تا پنجره شکل ۶-۲۸ نمایش داده شود. اکنون باید عکس پوشش (حامل) را مانند شکل ۶-۲۹ انتخاب کنیم.



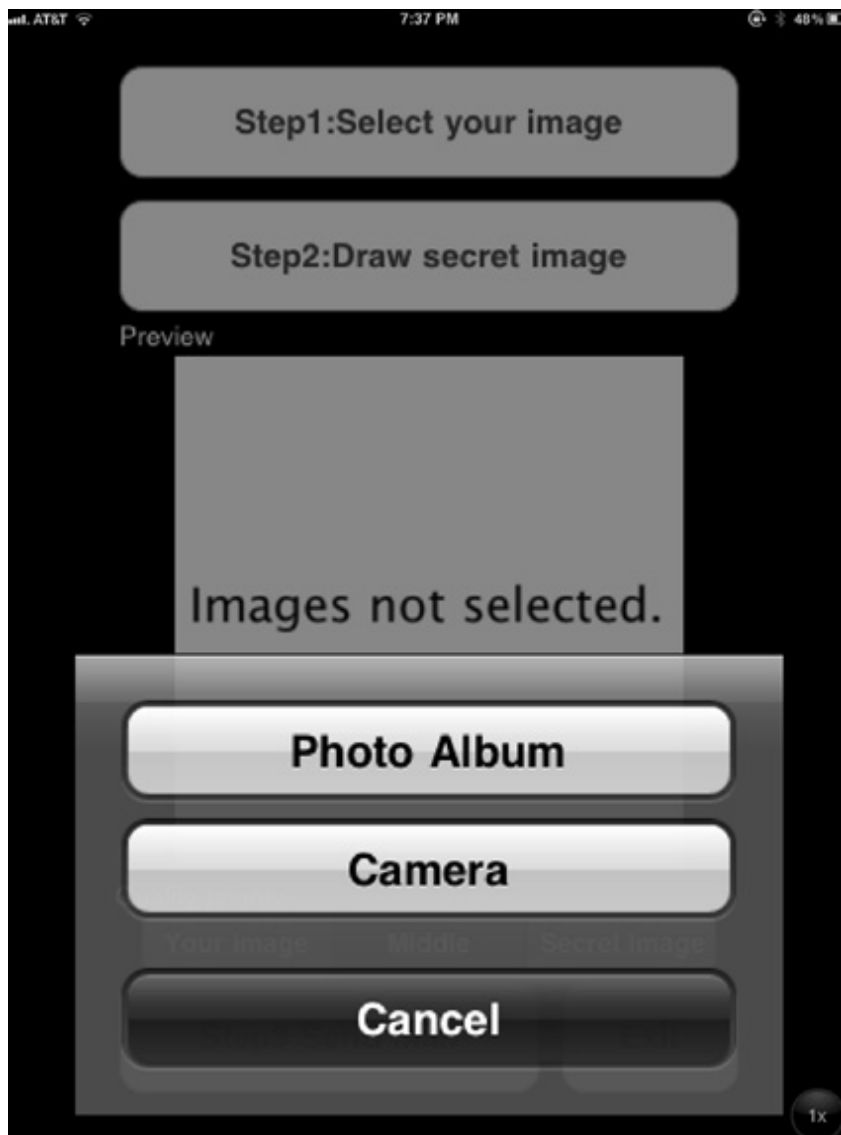
شکل ۶-۲۷: پنجره راهنمای نرم‌افزار InvisiLetter

| InvisiLetter DetailsX |                |
|-----------------------|----------------|
| Application Name      | InvisiLetter   |
| Seller                | Hideaki Tamori |
| Image format          | True color PNG |
| Last release          | July 2010      |

همان‌گونه که ممکن است حدس زده باشید، می‌توانید از عکس‌های موجود در گالری خودتان هم استفاده کرده یا عکس جدید بگیرید.



شکل ۶-۲۸: انتخاب عکس پوششی در Invisiletter

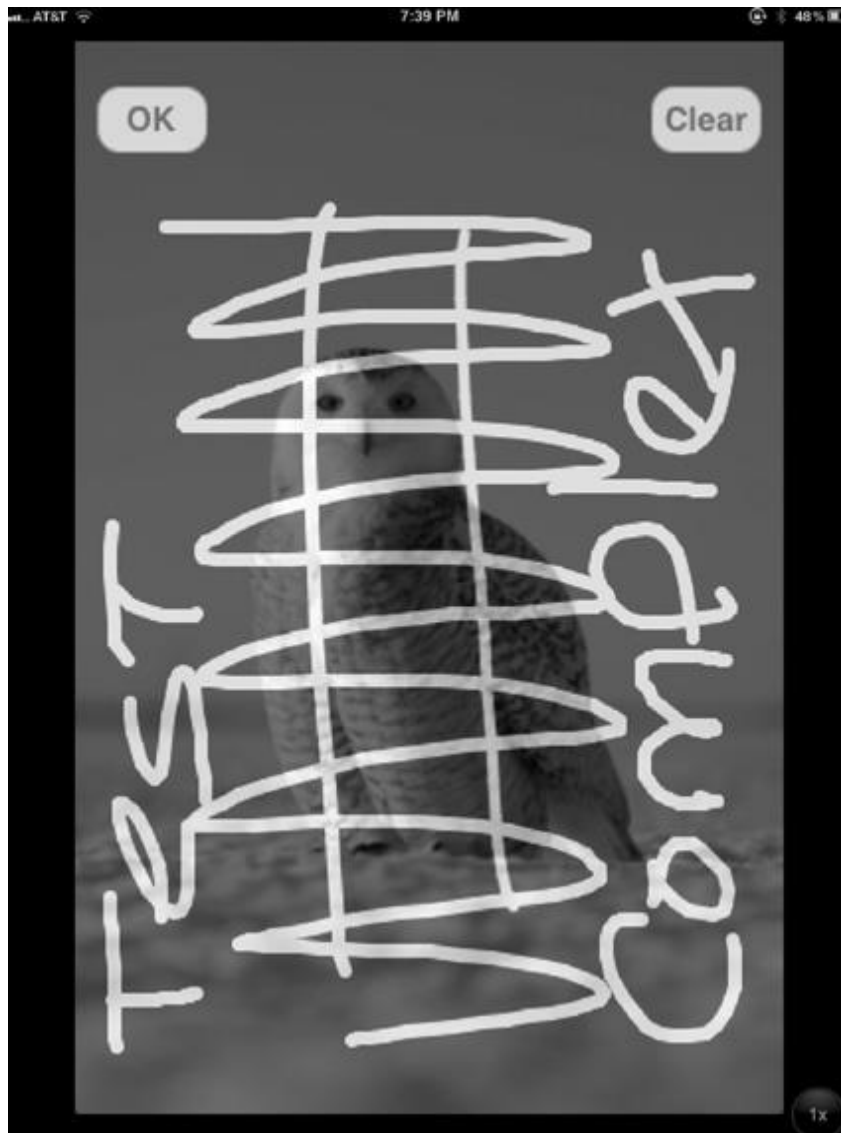


شکل ۶-۲۹: گزینه انتخاب عکس پوششی با عکس جدید یا عکس موجود

پس از انتخاب عکس پوشش App، امکان اضافه کردن پیام بر روی عکس به وسیله‌ی تایپ یا قلم یا هر وسیله مناسب دیگر را به شما می‌دهد. در این مثال برای توضیح روش پنهان‌سازی، متن ساده و متن کمی پیچیده‌تر را در نظر می‌گیریم (شکل ۶-۳۰ و ۶-۳۱).



شکل ۶-۳۰: افزودن پیام ساده در Invisiletter

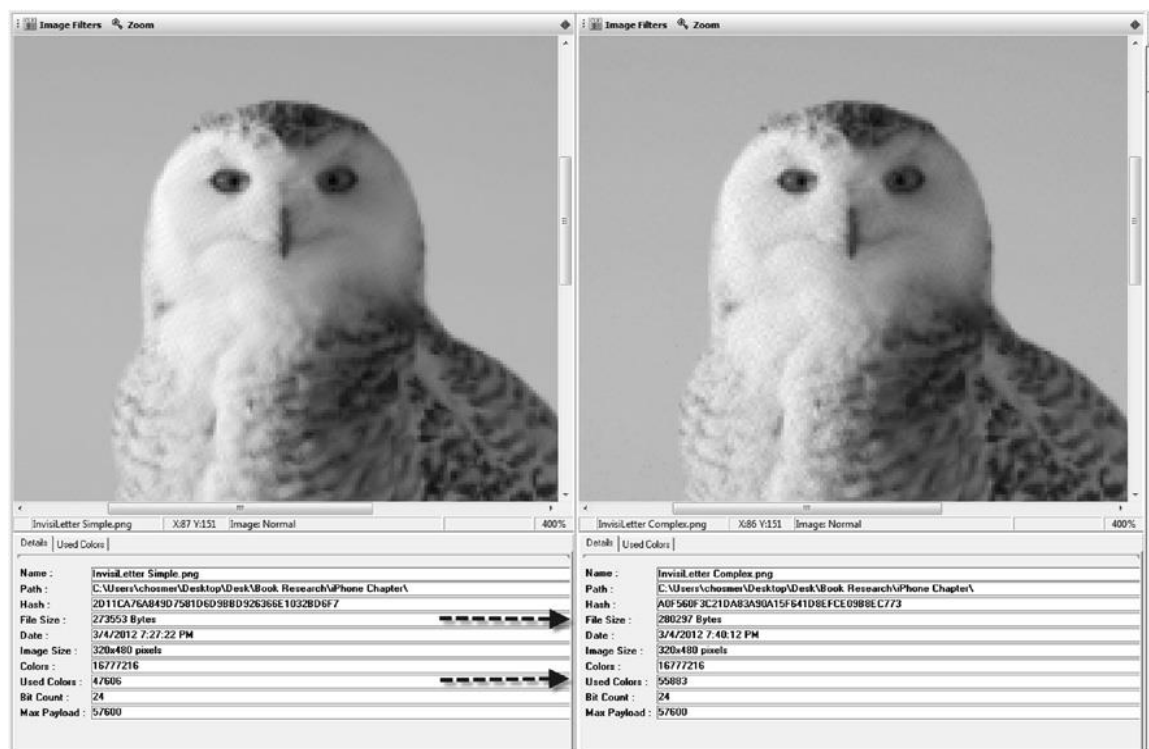


شکل ۶-۳۱: افزودن پیام پیچیده در Invisiletter

## تحلیل روش پنهان‌سازی داده

تحلیل این روش نهان‌سازی نیازمند تحلیل با اندکی تفاوت با روش تحلیل‌های پیشین است و از آنجایی که عکس حاصل از نوع PNG است که شامل پیام پنهانی نیز می‌باشد، باید به تفاوت موجود در دو تصویر توجه کنیم. همان‌گونه که در تحلیل Stego set فقط پیام پنهان شده را تغییر دادیم، این بار نیز تنها پیام را عوض می‌کنیم. در شکل ۶-۳۲ پیام ساده را در سمت چپ و پیام پیچیده را در سمت راست نشان داده‌ایم.





شکل ۳۲-۶: مقایسه تغییرات حاصل از افزودن پیام ساده و پیام پیچیده

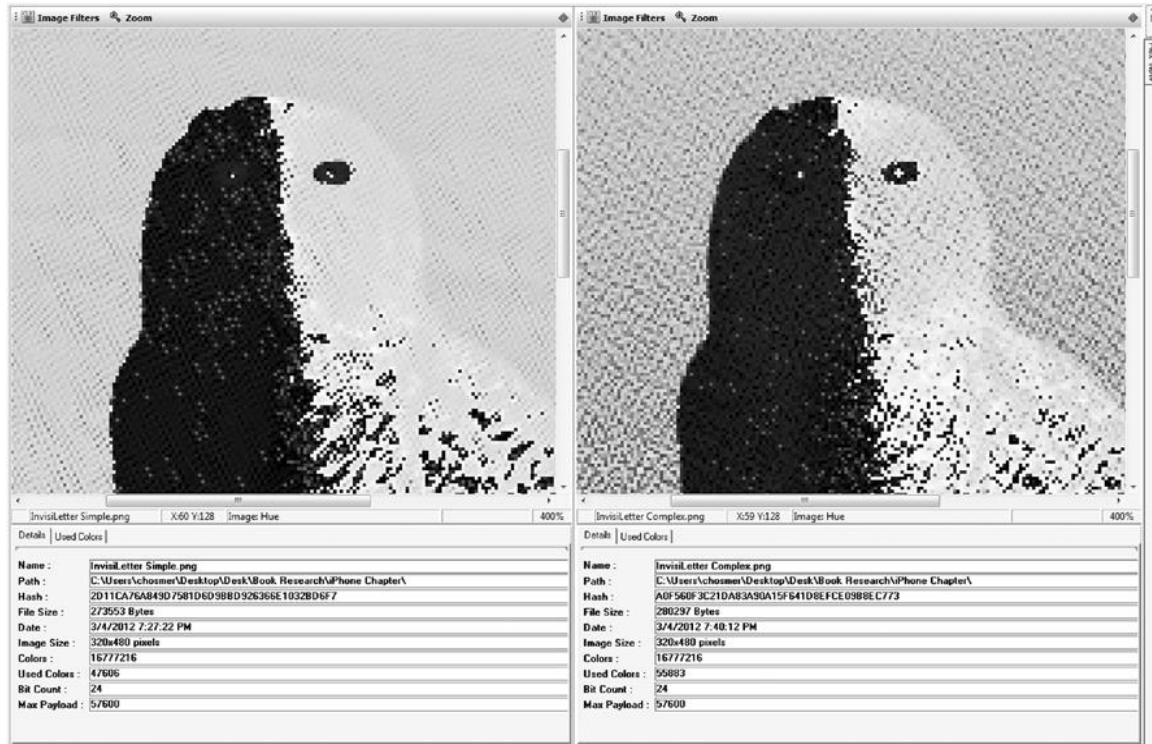
در نگاه اول دو تفاوت چشمگیر بین پیام ساده و پیام پیچیده وجود دارد.

(۱) فایل حاوی پیام کوتاه اندازه‌ی کمتر از ۶۷۴ بایت دارد و این خیلی عجیب نیست، زیرا این اطلاعات اضافی می‌بایست به صورت پنهان اضافه شود تا داده‌ها و کلمات اضافه را ذخیره کند. بخش IDAT عکس در قالب PNG فشرده می‌شوند. با این حال هر دستکاری در مقادیر RGB پیش از فشرده سازی، قابلیت فشرده سازی را تغییر می‌دهد.

(۲) افزایش اندازه‌ی فایل بیانگر تعداد بیشتر رنگ‌های مورد استفاده در عکس پیچیده است. ۵۵۸ در برابر ۴۱۶ رنگ. هرگاه این سبک از افزایش در رنگ‌های استفاده شده (برای فایل‌های حامل یکسان) را دیدیم، دلالت بر تغییر مقادیر کم ارزش‌ترین بیت دارد زیرا خود این افزایش باعث افزایش تعداد رنگ‌های جداگانه‌ای است که در عکس پیچیده‌ی فشرده نشده می‌توان یافت.

با نگاه به عکس نمی‌توان تفاوت آشکاری به شکل اعوجاج یا بخشی با داده‌های ساختگی در فایل عکس مشاهده کرد. این عکس ۴۰٪ زوم شده که به صورت لبه‌های دنداندار دیده می‌شود که در مورد عکس اصلی، عکس با پیام ساده و عکس با پیام پیچیده صدق می‌کند.

برای مشاهده‌ی تفاوت‌ها بین عکس ساده و پیچیده و تعیین تغییراتی که پنهان‌سازی داده‌ها مسبب آن‌ها بوده، باید عکس را به شکل دیگری پردازش کنیم. در عکس ۶-۳۳ تصمیم گرفتیم فام هر عکس را رندر کنیم.



شکل ۶-۳۳: تحلیل InvisiLetter پیام ساده و پیچیده در نرم‌افزار Stego Analyst

تفاوت‌ها با افزودن داده‌های پنهانی بیشتر و بیشتر، به تدریج آشکار می‌گردد که بیانگر نمادی از داده‌های پنهان شده است که در بیت‌های RGB عکس جاسازی شده‌اند. با نگاهی به عکس اصلی مشخص می‌شود که تمام عناصر ساختاری عکس دستکاری نشده و تغییر نکرده‌اند؛ پس پی می‌بریم که جاسازی داده‌ها با تغییر مقادیر RGB رنگ‌ها در عکس انجام گرفته است.



شکل ۶-۳۴: برخی از نرم افزارهای پر کاربرد پنهان سازی در Ipad App

## چکیده

در حال حاضر بسیاری از نرم افزارهای پنهان سازی داده ها برای Ipad وجود دارد و بسیاری دیگر هم هر روز تولید شده یا عملکردشان بهبود می یابد. پیشرفت سریع در ارائه امکانات و طیف گسترده ی روش ها و فن های به کار رفته در آن ها، بسیاری از ما را تا پاسی از شب برای توسعه ی روش های جدید تحلیل، تشخیص، ره گیری و کاهش تأثیرات بالقوه ویرانگر در کاربردهای نابکارانه، به کار مشغول نگه می دارد و تقاضای بازار برای ارائه ی راه های بهتر پنهان سازی داده ها و برقراری ارتباط پوشیده، افزایش هشدار دهنده ای دارد.

این گفته چه پیامی دارد؟

(۱) آیا مردم تصور می کنند ویژگی های رمزنگاری ابزارهای مورد استفاده آن ها در خور و شایسته هستند؟

(۲) آیا مردم به APP رمزنگاری تجاری اعتماد ندارند؟

~~~~~

۳) آیا مردم بیشتر به پنهان کردن ارتباط موجود توجه می‌کنند تا به حفظ حریم خصوصی داده-

هایشان؟

۴) آیا مردم انتظار حافظت بهتری از اطلاعات شخصیشان را دارند؟

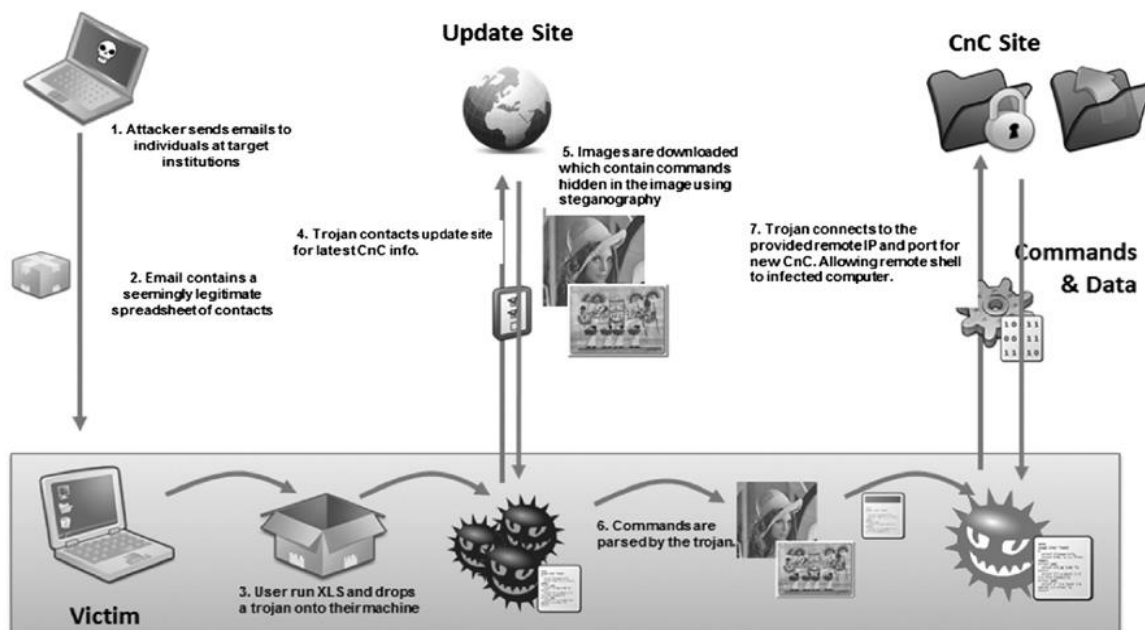
## فصل هفتم

### پنهان سازی داده ها در سیستم عامل ویندوز و لینوکس

ویندوز همچنان سیستم عامل فراگیر در رایانه های خانگی است. بنابراین عجیب نیست اگر هدف اکثر بدافزارها هم باشد. بدافزارها برای گریز از کشف شدن، پیچیده تر شده اند. مثلاً برخی نرم افزارها امکان ساختار ماژولار را دارند و این رویکرد ماژولار به بدافزارها امکان حذف برخی اجزا و افزودن اجزا دیگر را به گونه ای می دهد که بدافزار دیگری خلق کنند. گاهی بدافزارها ممکن است به فایلی در نشانی وب سایت بی ضرری مثل Wordpress اشاره کند ولی در اصل ممکن است حاوی فایلی با آدرس IP دیگری برای دسترسی، کنترل و ارسال دستورهای اجرایی باشد. حتی شاید این آدرس، برای ناکام گذاشتن تشخیص و کشف بدافزار به صورت مرتب به روزرسانی می شود؛ بدافزار خود را به سایت مربوطه برای دریافت فایلی با آخرین فرمان ها کنترل دستوری می رساند. از این گذشته برخی از دستورات ممکن است در فایل های تصویری با استفاده از روش های استتار جاسازی شده باشند. با ترکیب چند روش؛ طراح بدافزار می تواند نرم افزار گریزان از شناسایی ایجاد کند درحالی که می تواند همزمان از امکان جایگزینی اجزا و تغییر در ساختار و ارتقاء عملکرد هم استفاده نماید. یک مثال از این نمونه در دنیای واقعی عملیات shady RAT است.

عملیات shady RAT از روش های گوناگونی استفاده می کند، ولی پرکاربردترین شکل آن در سیستم عامل ویندوز استفاده از لیست نشانی ها و پست الکترونیکی است که بدافزار به شکل ضمیمه به آن می پیوندد (عکس ۷-۱). روش حمله به شرح زیر است:

- ۱) مهاجم اقدام به ارسال پست الکترونیکی شناسایی هدف می کند (موسوم به نیزه ماهیگیری).
- ۲) این پست الکترونیکی حاوی فایلی از نشانی ها و تماس هایی است که به نظر قانونی و بی اشکال می رسد.



شکل ۷-۱: جزئیات چگونگی اجرای عملیات Shady Rat

(۳) کاربر فایل صفحه‌ی گستره‌ی پیوست نامه را باز کرده و ناخواسته اسب تراوا را به رایانه خود منتقل می‌کند.

(۴) اسب تراوا به نشانی سایت به ظاهر بی‌ضرر، مثل یک صفحه‌ی تصادفی از سایت Wordpress، برای دریافت اطلاعات و آخرین دستورات کنترلی و فرمان‌ها اجرایی ارتباط برقرار می‌کند.

(۵) به جای دریافت مستقیم اطلاعات از سایت، اسب تراوا اقدام به دریافت فایل‌های تصویری می‌کند که حاوی دستورات پنهان شده به روش‌های استتار در آن فایل است.

(۶) این دستورات به وسیله‌ی اسب تراوا تجزیه‌شده تا آدرس IP و شماره پورت و... سرور C&C را استخراج کند.

(۷) اسب تراوا به آدرس IP و درگاه دریافتی متصل شده و دسترسی به رایانه را از راه دور فراهم می‌کند.

این روش برای بار اول به وسیله‌ی شرکت Mcafee پس از ۵ سال پژوهش در فراهمایی Black Hal در لاووکای اعلام شد. این پژوهش از اوایل سال ۲۰۰۶ میلادی تا اواسط سال ۲۰۱۱ میلادی تا پیش از انتشار گزارش ادامه داشت. بر اساس این گزارش، عملیات RAT shady71 شرکت چندملیتی، مؤسسات دولتی و سازمان‌های غیرانتفاعی را درگیر و متأثر کرد و این مؤسسات شامل دولت فدرال آمریکا، سازمان ملل متحد، پیمانکاران نظامی و شرکت‌های تولید برق و شرکت‌های فناوریانه بودند. به علاوه کشورهای که مورد حمله قرار گرفتند شامل کانادا، کره جنوبی، ژاپن، آلمان و بسیاری دیگر بودند.

کشف تکان دهنده دیگری، مدت زمانی بود که این کشورها هدف این نوع حمله بودند که از چند ماه تا چند سال متغیر بود. نیاز به گفتن نیست که پیامدهای آن در آینده مشخص می شود.

کمی پس از انتشار این گزارش توسط شرکت مک آفی، شرکت سیمانتک، لایه ای دیگر از حمله که در گزارش اولیه به آن اشاره نشده بود را اعلام کرد. سیمانتک جزئیات فنی بیشتر از چگونگی عملکرد Shady Rat و چگونگی ذخیره اطلاعات در سرور CNC به وسیله ی فایل های تصویری را با استفاده از روش های استتار منتشر کرد. از آنجایی که فایروال ها و سیستم های تشخیص نفوذ مهاجم IDS به فایل های عکس اجازه عبور می دهند پس این فایل ها به راحتی توسط اسب تراوا برای استخراج آخرین اطلاعات دستورات کنترلی و فایل های اجرایی دانلود می شد. با استفاده از این روش، کاملاً امکان پذیر است که بسیاری از محصولات امنیتی آخرین لیست CNC در اینترنت را نداشته باشند و در عمل، یک راه مؤثر برای اسب تراوا را فراهم می کرد که یک گام جلوتر از کاشفان CNC باشد.

اسب تراوا دیگری که از روش های چندبعدی عملکرد اسب تراوا تحت ویندوز که از روش های استتار برای پنهان کردن دستورات در داخل متن استفاده می کرد، Alurean-Trojan است. این روش و روش های نوین دیگر نشان دادند که رخنه گرها چند روش را در یک بدافزار پیاده می کنند تا به آن قدرت عمل اضافی داده و اسب تراوا را از کشف شدن در امان نگه دارند. در این فصل چند روش تازه در پنهان سازی داده ها در سیستم عامل ویندوز را بررسی می کنیم.

## پنهان سازی داده ها در سیستم عامل ویندوز

### مروری بر Alternate Data streams

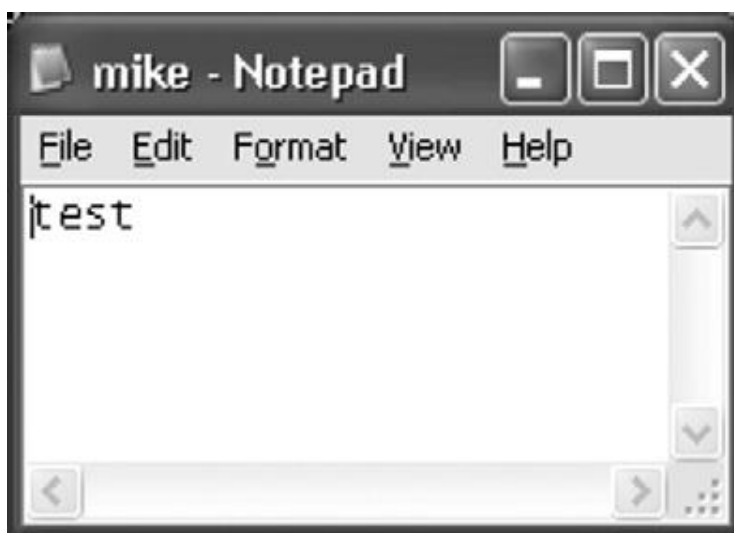
Alternate Data streams از زمان ارائه ویندوز ۳,۱ شناخته شده بوده و هدف اصلی از طراحی آن سازگاری با فایل سیستم سلسله مراتبی<sup>۱</sup> سیستم فایل سلسله مراتب سیستم عامل مکتناش بود. فایل سیستم NIFS از Alternate Data streams برای ذخیره ابر داده های مرتبط با فایل که شامل اطلاعات امنیتی، کاربر ایجاد کننده فایل و سایر ابر داده ها استفاده می کند.

Alternate Data streams در فایل سیستم NT راه ساده و مؤثر برای پنهان کردن فایل های حامل داده های پنهانی است، زیرا جستجو در پوشه ی فایل ها، چیزی بیش از فایل های مورد انتظار موجود در پوشه را آشکار نمی کند. تا زمانی که همه چیز عادی به نظر برسد، فایل های پنهان شده به وسیله

<sup>۱</sup> Hierarchical File System

Alternate Data streams کشف نشده باقی می ماند. در مثال زیر چگونگی بکارگیری Alternate Data streams برای پنهان کردن یک یا چند فایل را در ویندوز شرح می دهیم. این روش سازوکار ساده و ناپدید از دید فایل سیستم ویندوز برای نهان سازی فایل ها را فراهم می نماید. برای شروع فایل، متنی ساده به نام mike.txt را ایجاد می کنیم.

D:\mike>notepad mike.txt



شکل ۷-۲: ایجاد فایل متنی mike.txt به وسیله نرم افزار notepad

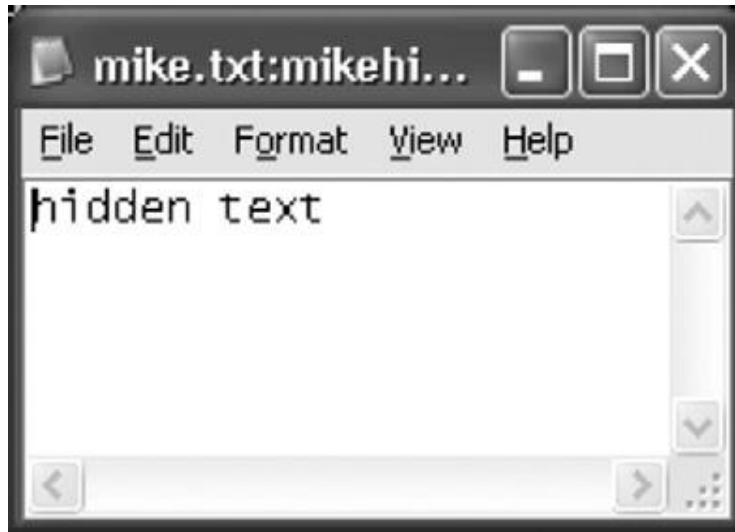
البته حالا می توانیم با لیست گرفتن از محتویات پوشه، فایل تازه ایجاد شده را به شکل زیر مشاهده نماییم.

D:\mike>dir

```
Volume in drive D has no label.
Volume Serial Number is FFFF-FFFF
Directory of D:\mike
11/07/2005 07:17 PM    <DIR>        .
11/07/2005 07:17 PM    <DIR>        ..
11/07/2005 07:17 PM                4 mike.txt
    1 File(s) 4 bytes
    2 Dir(s) 1,029,111,808 bytes free
```

اکنون با استفاده از فایل ایجاد شده، فایل پنهان از دید سیستم عامل را با Alternate Data streams به شکل زیر ایجاد می کنیم (عکس ۷-۳).





شکل ۷-۳: ایجاد Alternate Data Stream

D:\mike>notepad mike.txt:mikehidden.txt

فایل ذخیره شده به وسیلهی Alternate Data streams با استفاده از روش های عادی لیست گیری از پوشه ها نمایش داده نمی شود. لیست گیری به وسیلهی دستور از خط فرمان ویندوز یا به وسیلهی ویندوز Explorer هیچ نشانه ای از وجود فایل جدیدی را نشان نمی دهد؛ حتی اندازه ی پوشه و فضای خالی باقیمانده بر روی بر دیسک هم عوض نمی شود و علی رغم ایجاد فایل هیچ مدرکی دال بر وجود آن هم نشان داده نمی شود.

D:\mike>dir

Volume in drive D has no label

Volume Serial Number is FFFF-FFFF

Directory of D:\mike

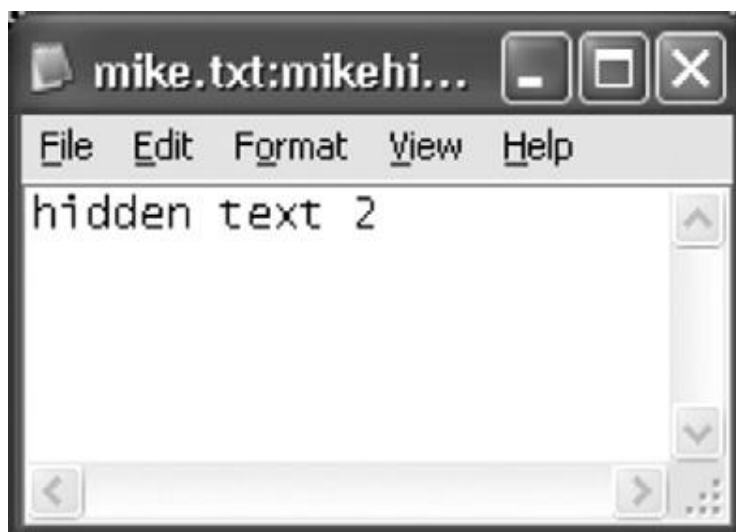
```

11/07/2005 07:17 PM    <DIR>    .
11/07/2005 07:17 PM    <DIR>    ..
11/07/2005 07:18 PM                4 mike.txt
1 File(s)                4 bytes
2 Dir(s)      1,029,111,808 bytes free

```

نکته شایان توجه اینکه حتی محدود به یک Alternate Data streams به ازای هر فایل نبوده

و می توانیم چندین ADS گوناگون را به فایل mike.txt پیوند بزنیم (عکس ۷-۴).



شکل ۷-۴: پنهان سازی دیگری در ADS در فایل mike.txt

D:\mike>notepad mike.txt:mikehidden2.txt

بار دیگر می توانیم از پوشه لیست گرفته و مشاهده کنیم که هیچ نشانه ای دال بر وجود مدرکی از فایل ایجاد شده در ADS در پوشه ها وجود ندارد.

```
D:\mike>dir
Volume in drive D has no label.
Volume Serial Number is FFFF-FFFF
Directory of D:\mike
11/07/2005 07:17 PM <DIR> .
11/07/2005 07:17 PM <DIR> ..
11/07/2005 07:18 PM      4 mike.txt
1 File(s)      4 bytes
2 Dir(s)      1,029,111,808 bytes free
```

شایان توجه است که بیشتر بسته های ویروس کش به صورت پیش فرض Alternate Data streams ویندوز را برای وجود اسب تراوا و سایر کدهای مخرب جستجو نمی کنند. اگر شما بازرسی تحقیقات قضایی انجام می دهید، مطمئن شوید که از این ویژگی بسیار مهم در ویروس کش استفاده نمایید. اگر این ویژگی توسط نرم افزار ویروس کش پشتیبانی می شود، آن را برای بررسی خود فعال نمایید. عیب این کار این است که فعال سازی این گزینه باعث افت سرعت ویروس کش به میزان ۱/۱۰ می شود. به همین علت بسیاری از شرکت های تولید کننده ویروس کش این گزینه را به طور پیش فرض غیر فعال

می کنند. لب کلام این که Alternate Data streams معمولاً در بررسی ها نادیده گرفته می شود و بنابراین می تواند مکان مناسبی برای پنهان سازی فایل ها باشد.

## Stealth Alternate Data stream

راه های نامرئی دیگری برای پنهان کردن در Alternate Data streams وجود دارد. یکی از آنها ضمیمه کردن Alternate Data streams به نام Device رزرو شده در سیستم عامل است، Alternate Data stream حتی ابزارهایی همچون LDS و STREAM.txt غیرقابل تشخیص می کند ویندوز شامل چندین نام device رزرو شده است که از آنها نمی توان به عنوان نام فایل استفاده کرد. سایت MSDN نام های رزرو را به شرح CON, PRN, AUX, NUL, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9 اعلام کرده است. این اسامی رزرو شده تحت عنوان خروجی به ابزارهای سخت افزاری اختصاص داده شده اند. با کمی دقت در سایت متوجه می شویم که کلمه کلیدی "نباید" است که دلالت بر این نکته دارد که امکان اختصاص آنها وجود دارد. اما نباید این اختصاص انجام شود. در مثال زیر نخست فایلی را با استفاده از دستور echo ایجاد کرده، سپس با همین دستور سعی می کنیم فایلی با استفاده از کلمات کلیدی به شرح زیر ایجاد کنیم:

```
C:\sandbox>echo mike > mike.txt
C:\sandbox>echo mike > COM1.txt
The system cannot find the file specified.
C:\sandbox>mkdir COM1
The directory name is invalid.
C:\sandbox>dir
Volume in drive C has no label.
Volume Serial Number is AAAA-BBBB
Directory of C:\sandbox
02/28/2012  03:21 PM          <DIR>          .
02/28/2012  03:21 PM          <DIR>          ..
02/28/2012  03:21 PM              7 mike.txt
                1 File(s)              7 bytes
                2 Dir(s)          198,873,174,016 bytes free
```

همان گونه که مشاهده می کنید تلاش برای ایجاد فایل یا پوشه ای تحت نام رزرو شده باعث بروز خطا می شود، ولی ترفندی برای گذر از این محدودیت وجود دارد.

از آنجایی که دلیل اولیه اختصاص کلمات کلیدی به ابزارهای سخت افزاری، مقاصد مورد نظر عملیات I/O بوده، پس برای ایجاد فایل با این اسامی رزرو شده می بایست بدون استفاده از تجزیه گر استاندارد، نام فایل انتخابی را مستقیم به فایل سیستم ارسال کنیم. با غیرفعال کردن تجزیه گر رشته اجازه ارسال مستقیم نام فایل به فایل سیستم فراهم می شود. این راهکار را برنامه نویسان سیستم برای کار با API ویندوز به کار می برند، ولی ما از این راهکار برای ایجاد فایل هم نام device رزرو شده استفاده می کنیم. با ترکیب اسامی رزرو شده device و پیشوند \\?\\، می توانیم از تجزیه گر استاندارد نام فایل گذر کرده و فایلی تحت نام رزرو در سیستم عامل ویندوز ایجاد کنیم.

در مثال زیر با استفاده از \\?\\ و ترکیب آن با نام رزرو شده فایلی تحت عنوان Null به شکل زیر ایجاد می کنیم.

```
C:\sandbox>echo mike > \\?c:\sandbox\NUL
```

لیست گرفتن از پوشه، فایل تازه ایجاد شده را نشان می دهد و اگر بخواهیم از تحلیل هایی با مقاصد قضایی یا اسکن به وسیله ی ویروس کش، گذر انجام دهیم این ابزار می تواند مفید باشد.

```
C:\sandbox>dir
Directory of C:\sandbox
01/25/2012      09:13 PM                <DIR>      .
01/25/2012      09:13 PM                <DIR>      ..
01/25/2012      09:15 PM                7 NUL

C:\sandbox>more NUL
Cannot access file \\.\NUL
```

کاربر آشنا با این روش می تواند محتویات فایل را با استفاده از دستور \\?\\ به شکل زیر مشاهده نماید.

```
C:\sandbox>more \\?c:\sandbox\NUL
mike
```

می توان با ترکیب این فن با stealth Alternate data stream روش نوینی از پنهان سازی داده ها را خلق می کرد.

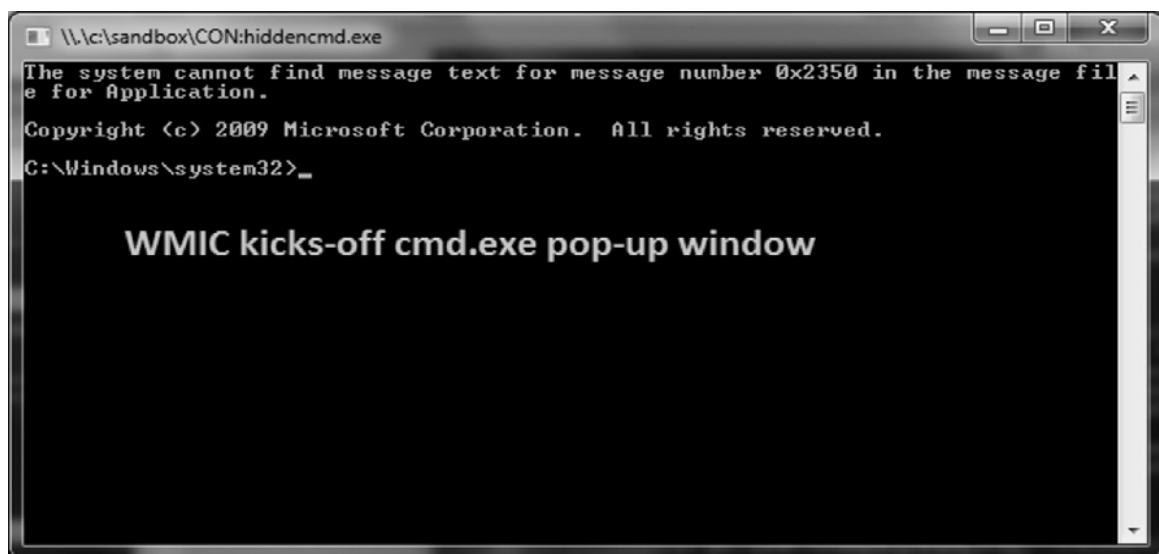
برای نامرئی شدن بیشتر فایل از دید سیستم عامل ویندوز می توانیم نام های رزرو device را با stealth Alternate data stream ترکیبی و stealth Alternate data stream را درست کنیم. چند مزیت در این روش وجود دارد؛ نخست: stealth Alternate data stream به وسیله ی ابزارهای stealth Alternate data stream مثل stream.exe dir /r و سایر تکنیک ها قابل

تشخیص نیست؛ به علاوه بسیاری از ابزارها به قدر کافی قدر نیستند تا stealth Alternate data stream را تشخیص دهند. دوم: اگر پنهان سازی فایل در stealth Alternate data stream به شکل فایل اجرایی باشد و بتوان با استفاده از Wmic (windows Management Instrumentation command line) آن را اجرا کرد در ترکیب با windows powershell راه مؤثری برای پنهان سازی و اجرای بدافزارها می شود.

در این مثال برنامه CMD.exe را به نام Con و با استفاده از stealth Alternate data stream برای ایجاد stealth Alternate data stream به شکل زیر پیوند می زنیم.

```
C:\sandbox> type cmd.exe > \\.\c:\sandbox\CON:hidecmd.exe
```

می توانیم از WMIC برای اجرای CMD.exe که در stealth Alternate data stream پنهان شده استفاده کنیم که اجرای برنامه Cmd در قالب پنجره ای در ویندوز می گردد. اگرچه اجرای این دستور چندان بدخواهانه به نظر نمی رسد، اما قابلیت بی پایان این روش در نهان کاری را آشکار می کند (عکس ۷-۵).



شکل ۷-۵: پنجره نمایش داده شده به وسیله WMIC

```
C:\sandbox> wmic process call create  
\\.\c:\sandbox\CON:hidecmd.exe
```

```
Executing (Win32_Process)->Create()
```

```
Method execution successful.
```

```
Out Parameters:
```

```
instance of __PARAMETERS
```

```
{
```

```
    ProcessId = 8696;
```

```
    ReturnValue = 0;
```

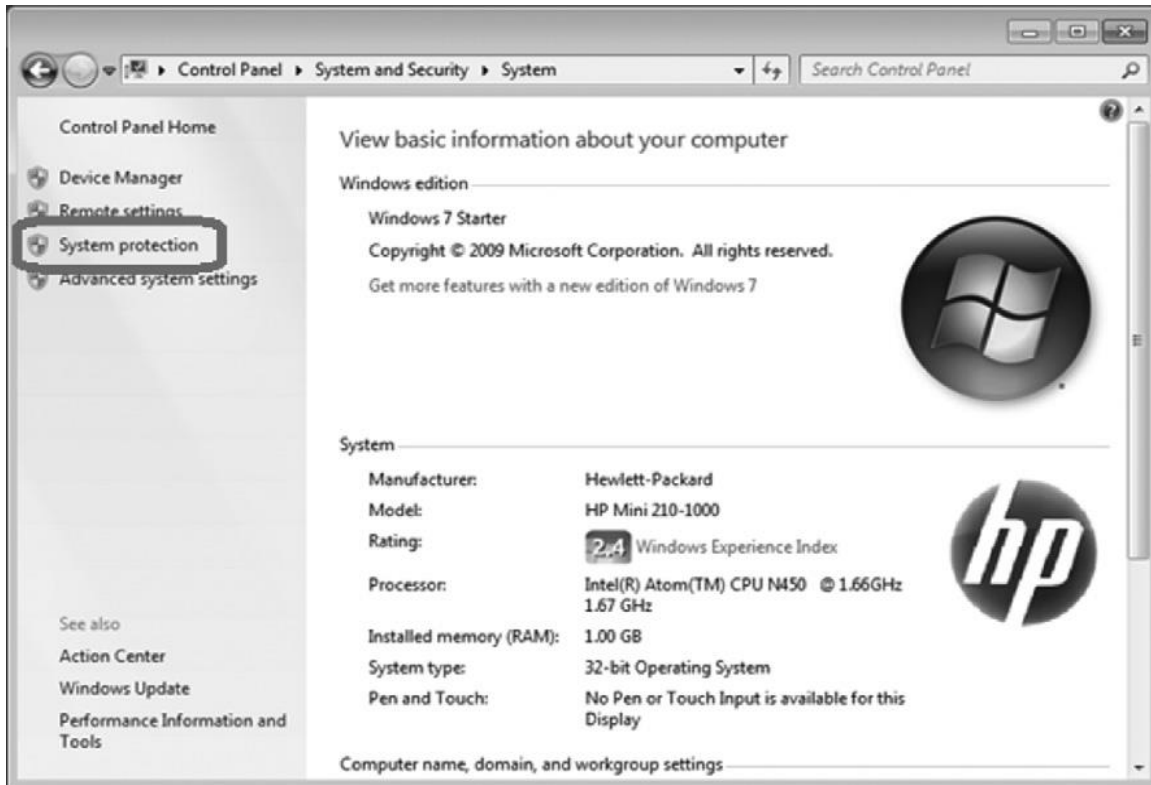
```
};
```

ویروس کش ها stealth Alternate data stream را تشخیص نداده حتی Alternate data stream را برای یافتن بدافزار هم جستجو نمی کنند.

## Volume Shadow

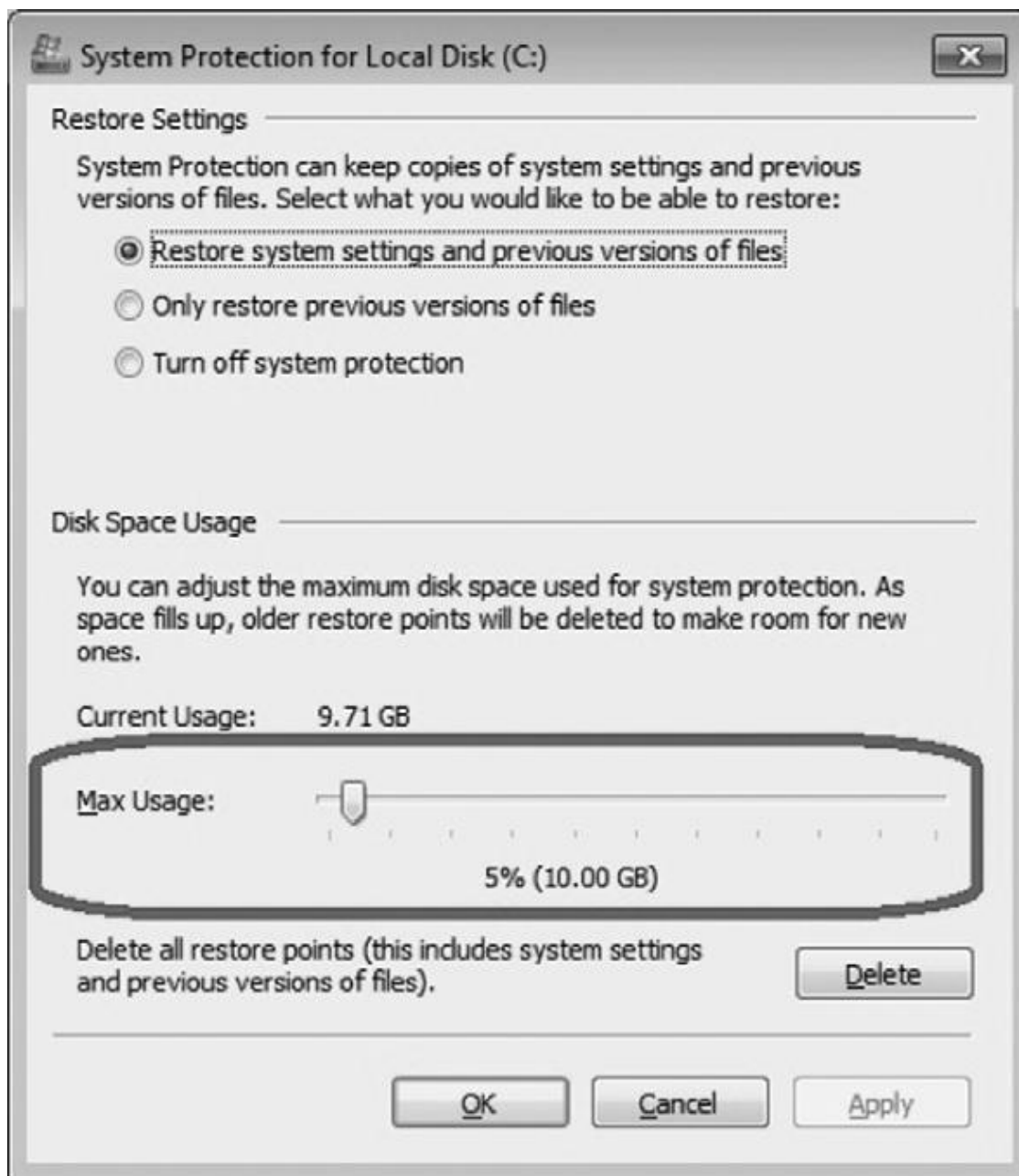
نسخه های جدید ویندوز مثل ویندوز ویستا و ۷، سرویس Volume Shadow Copy دارند و زمانی که نرم افزار یا درایوری نصب شود یا برنامه ای Crash کند، از داده ها پشتیبان تهیه می کند. این پشتیبان گیری به صورت داخلی انجام شده و از سیستمی به سیستم دیگر متفاوت است و بستگی به زمان بیکاری سیستم و روش های پشتیبان گیری پیش از نصب نرم افزار و غیره دارد. قاعده کلی پشتیبان گیری هر یک یا دو روز یک بار و در ویندوز ۷ هر ۷-۸ روز یکبار است ولی در نظر داشته باشید مدت زمان بیکاری سیستم و کثرت نصب نرم افزارها بر قاعده بالا اثر می گذارند.

نکته مهم دیگری که باید در مورد سرویس Volume Shadow Copy شایان توجه است این نکته است که تمام نسخه های مختلف یک فایل را ذخیره نمی کند و عملکرد مشابه سیستم های X/VAX و mac osx lion دارد؛ به عبارت دیگر اگر فایل متنی را ویرایش کنید تمام نسخه های آن فایل را ذخیره نمی کند، بلکه تنها نسخه ای که در هنگام اعمال آخرین تغییرات ذخیره شده را پشتیبان گیری می کند؛ به علاوه تمام فایل ها هم پشتیبان گیری نشده و تنها فایل هایی که تغییر کرده اند پشتیبان گیری می شوند. Volume Shadow Copy بخشی از Volume یا هارد را برای این کار اختصاص می دهد. برای مشاهده ی پیکربندی Volume Shadow Copy از Control panel، گزینه System و سپس System Protection را انتخاب کنید (عکس ۷-۶).



شکل ۷-۶: دسترسی و مشاهده Volume copies

System Protection را برای مشاهده System Properties انتخاب کنید. Protection پیکربندی Shadow Copy volumes را نمایش می دهد و با کلیک دکمه ی Configure می توان اندازه فضای ذخیره سازی را مشاهده نموده و تغییر داد. مقدار بیشینه پیش فرض در ویندوز ویستا ۱۵٪ و در ۷ به اندازه ی ۵٪ اندازه هارد درایو می باشد و اما تنظیمات پیکربندی به شما اجازه افزایش آن را به مقدار بیشتر را هم می دهد. دقت کنید تغییرات ایجاد شده تنها بر سرویس Volume Shadow Copy تأثیر می گذارد و بر چگونگی پشتیبان گیری داخلی بی تأثیر است، بنابراین تغییرات به صورت تفاضلی با آخرین پشتیبان گیری قابل مشاهده است. در نتیجه سرویس Volume Shadow Copy از پشتیبان گیری افزایشی دقیقاً همانند پشتیبان گیری سرورها و بانک های اطلاعاتی استفاده می کند، پس کاملاً ممکن است که چند نسخه از فایل در Volume Shadow Copy وجود داشته باشد. به علاوه Volume Shadow Copy بر اساس اولویت بندی FIFO عمل می کند و هرگاه ظرفیت ذخیره سازی مشخص شده به پایان برسد، قدیمی ترین فایل آرشیو پاک می شود تا فضا برای آرشیو فایل جدید باز شود. نکته مهم دیگر این است که Volume Shadow Copy فایل فقط خواندی هستند (عکس ۷-۷).



شکل ۷-۷: پیکربندی Volume Shadow Copy

حالا که درک بهتری در مورد چگونگی کار Volume Shadow Copy داریم، اجازه دهید پتانسیل استفاده از این امکان را برای پنهان سازی داده ها بررسی کنیم. از آنجایی که بسیاری از ویروس کش ها Volume Shadow Copy را اسکن نمی کنند، پس مکان عالی برای پنهان کردن داده ها و



بدافزارهاست. ابزار VSSAD موجود در ویندوز ویستا و ویندوز ۷، به کاربر امکان مدیریت Volume Shadow Copy به وسیله ی خط فرمان<sup>۱</sup> را می دهد.

گزینه ی Last shadow امکان مشاهده آخرین Volume Shadow Copy را می دهد. آخرین فایل موجود در لیست لزوماً آخرین پشتیبان گرفته شده است.

```
C:\Windows\system32>vssadmin
```

```
C:\Windows\system32>vssadmin list volumes
```

```
vssadmin 1.1 - Volume Shadow Copy Service administrative command-
```

```
(C) Copyright 2001-2005 Microsoft Corp.
```

```
Volume path: \\?\Volume{33faab94-9bc6-11df-9987-806e6f6e6963}\
```

```
Volume name: \\?\Volume{33faab94-9bc6-11df-9987-806e6f6e6963}\
```

```
Volume path: C:\
```

```
Volume name: \\?\Volume{33faab95-9bc6-11df-9987-806e6f6e6963}\
```

```
Volume path: D:\
```

```
Volume name: \\?\Volume{33faab96-9bc6-11df-9987-806e6f6e6963}\
```

```
C:\Windows\system32>vssadmin list shadowstorage
```

```
vssadmin 1.1 - Volume Shadow Copy Service administrative command-  
line tool
```

```
(C) Copyright 2001-2005 Microsoft Corp.
```

```
Shadow Copy Storage association
```

```
For volume: (C:)\?\Volume{33faab95-9bc6-11df-9987-806e6f6e6963}\
```

```
Shadow Copy Storage volume: (C:)\?\Volume{33faab95-9bc6-11df-
```

```
9987-806e6f6e69
```

```
63}\
```

```
Used Shadow Copy Storage space: 9.707 GB (4%)
```

```
Allocated Shadow Copy Storage space: 9.94 GB (4%)
```

```
Maximum Shadow Copy Storage space: 10 GB (4%)
```

در این مرحله می خواهیم یک فایل را ایجاد و در Volume Shadow Copy پنهان کنیم. پس از آن چگونگی دسترسی به این فایل مستقل از فایل سیستم ویندوز را شرح می دهیم. در این مثال از فایل اجرایی Cmd.exe استفاده کرده و آن را به sandbox اضافه می کنیم.

---

<sup>۱</sup> Command line

```

C:\Windows\system32>vssadmin list volumes
vssadmin 1.1 - Volume Shadow Copy Service administrative command-line
(C) Copyright 2001-2005 Microsoft Corp.
Volume path: \\?\Volume{33faab94-9bc6-11df-9987-806e6f6e6963}\
    Volume name: \\?\Volume{33faab94-9bc6-11df-9987-806e6f6e6963}\
Volume path: C:\
    Volume name: \\?\Volume{33faab95-9bc6-11df-9987-806e6f6e6963}\
Volume path: D:\
    Volume name: \\?\Volume{33faab96-9bc6-11df-9987-806e6f6e6963}\

C:\Windows\system32>vssadmin list shadowstorage
vssadmin 1.1 - Volume Shadow Copy Service administrative command-
line tool
(C) Copyright 2001-2005 Microsoft Corp.

Shadow Copy Storage association
    For volume: (C:)\?\Volume{33faab95-9bc6-11df-9987-806e6f6e6963}\
    Shadow Copy Storage volume: (C:)\?\Volume{33faab95-9bc6-11df-
9987-806e6f6e69
63}\
    Used Shadow Copy Storage space: 9.707 GB (4%)
    Allocated Shadow Copy Storage space: 9.94 GB (4%)
    Maximum Shadow Copy Storage space: 10 GB (4%)

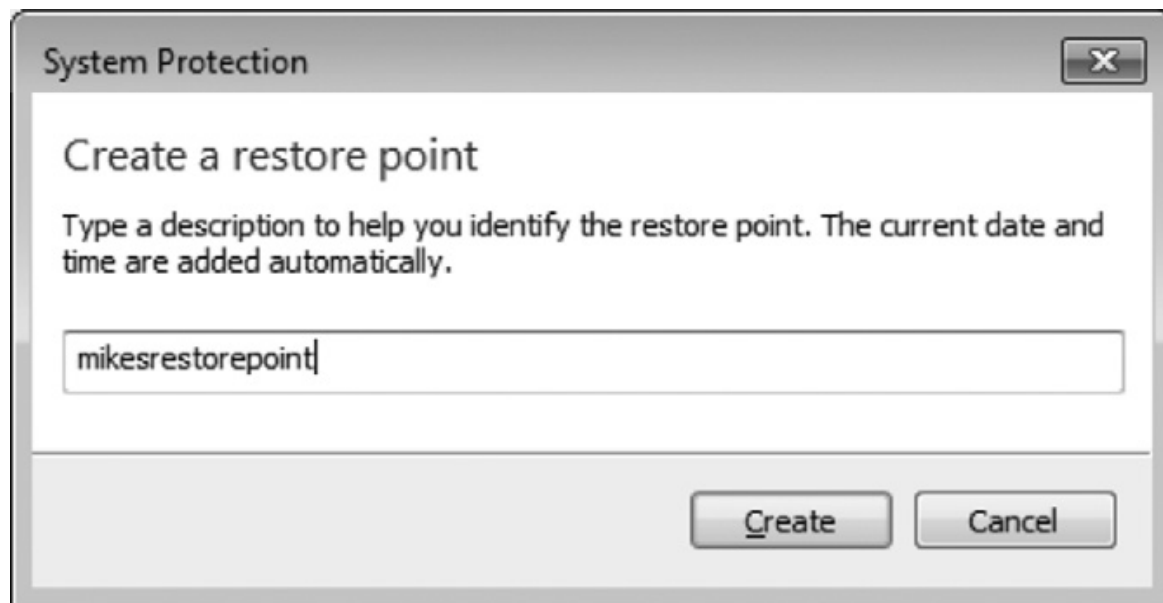
```

C:\sandbox>dir

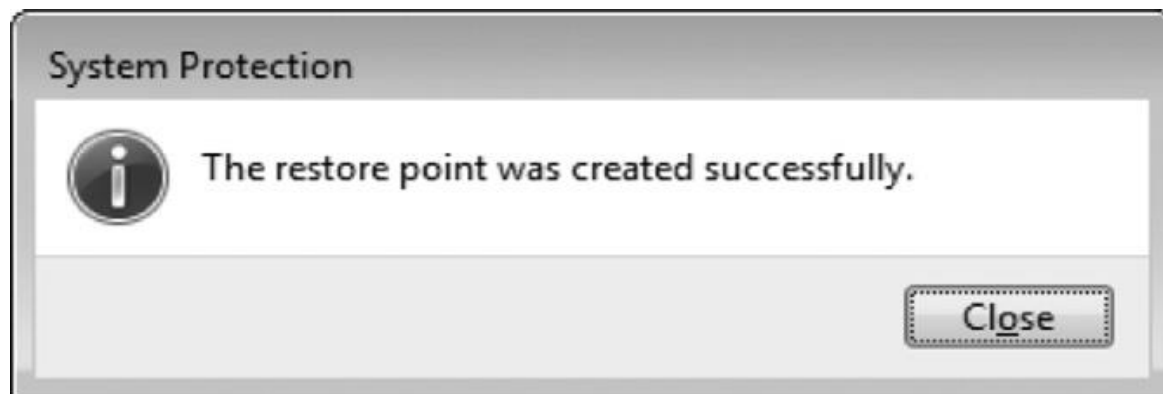
اکنون فایلی را که می‌خواهیم در Volume Shadow Copy پنهان کنیم را در اختیار داریم. restore point جدیدی ایجاد می‌کنیم که شامل فایل مورد نظر ما هم هست. برای این کار به System Properties رفته و گزینه‌ی Create را انتخاب کنید سپس نامی را که می‌خواهید به آن فایل ایجاد شود را وارد کنید (شکل ۷-۸ و ۷-۱۰).



شکل ۷-۸: ایجاد Restore Point



شکل ۷-۹: اختصاص نام به restore point



شکل ۷-۱۰: پیام ایجاد موفق restore point

برای تأیید ایجاد Volume Shadow Copy با استفاده از امکان vssadmin لیستی از Volume Shadow Copy گرفته و بررسی کنید که Volume Shadow Copy جدید در زمانی که آن را ایجاد کرده‌اید، در لیست وجود دارد یا نه؟

```
C:\Windows\system32> vssadmin list shadows
```

```
.
.
.
```

```
Contents of shadow copy set ID: {85e1aa26-d2d5-4ec5-88c5-2149b1f1f544}
```

```
Contained 1 shadow copies at creation time: 4/2012 5:41:00 PM
```

```
Shadow Copy ID: {19e1084c-7965-4092-9bf4-44dc55c1145a}
```

```
Original Volume: (C:\\?\\Volume{33faab95-9bc6-11df-9987-806e6f6e6963})\\
```

```
Shadow Copy Volume:
```

```
\\?\\GLOBALROOT\\Device\\HarddiskVolumeShadowCopy28
```

```
Originating Machine: funhouse
```

```
Service Machine: funhouse
```

```
Provider: 'Microsoft Software Shadow Copy provider 1.0'
```

```
Type: ClientAccessibleWriters
```

```
Attributes: Persistent, Client-accessible, No auto release, Differential, Auto recovered
```

در این مثال یک ورودی جدید را به عنوان نتیجه ایجاد restore point مشاهده می کنیم. حال می توانیم پشتیبانی از volume گرفته که حاوی فایل اجرایی مورد نظر بوده و فایل اصلی آن را حذف کنیم.

```
C:\sandbox> del cmd.exe
```

با این کار فایل اجرای تنها در Volume Shadow Copy وجود داشته و فقط با بررسی های قضایی در سطح سیستم می توان به وجود آن پی برد. حالا می خواهیم محتویات Volume Shadow Copy را با ایجاد پیوند سمبلیک مشاهده کنیم. برای این کار باید نگاهی دوباره به لیست نمایش داده شده به وسیله - ی vssadmin انداخته و به نام Volume Shadow Copy توجه کنیم؛ سپس به وسیله دستور Mklink با سویچ D پوشه ای را با پیوند نمادین **Error! Hyperlink reference not valid.** ایجاد می کنیم. دقت کنید که "|" را به آخر نام، برای ایجاد پیوند نمادین پیش نیاز VSC اضافه می کنید.

```

C:\sandbox>mklink /D hiddendirectory
\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy28\
symbolic link created for hiddendirectory <==>
\\?\GLOBALROOT\Device\Harddisk
VolumeShadowCopy28\

```

پس از ایجاد پیوند نمادین برای آزمودن درستی در دسترسی به فایل، به سادگی می‌توانید از پوشه مربوطه لیست گرفته و پیوند نمادین به نام "hiddendirectory" را مشاهده نمایید.

```

C:\sandbox>dir
Volume in drive C has no label.
Volume Serial Number is 98B1-9C5A

Directory of C:\sandbox

03/04/.2012  05:52 PM          <DIR>          .
03/04/2012   05:52 PM          <DIR>          ..
03/04/2012   05:52 PM          <SYMLINKD>      hiddendirectory
[\\?\GLOBALROOT\Device\Ha
rddiskVolumeShadowCopy28\]
02/28/2012   03:21 PM             7 mike.txt
               1 File(s)          7 bytes
               3 Dir(s)  199,002,898,432 bytes free

```

حال می‌توانیم به محتویات Volume Shadow Copy به وسیله‌ی پیوند نمادین دسترسی داشته و لیست محتویات را مشاهده کنیم. به علاوه امکان اجرایی بودن فایل Cmd.exe را که در Volume Shadow Copy پنهان کرده‌ایم را آزمایش کنیم.

```

C:\sandbox>cd hiddendirectory
C:\sandbox\hiddendirectory>dir
    Directory of C:\sandbox\hiddendirectory
11/10/2010  10:01 PM                1,024.rnd
06/10/2009  04:42 PM                24 autoexec.bat
03/04/2012  05:19 PM    <DIR>          book
06/10/2009  04:42 PM                10 config.sys
02/13/2012  04:38 PM    <DIR>          HP Universal Print Driver
03/04/2012  05:29 PM    <DIR>          myprogram
02/24/2012  02:45 PM          56,384 offreg.dll
07/13/2009  09:37 PM    <DIR>          PerfLogs
07/29/2011  02:31 PM    <DIR>          Personal
10/21/2010  10:17 AM    <DIR>          Pre
04/14/2011  10:10 PM    <DIR>          Program Files
03/04/2012  05:39 PM    <DIR>          sandbox
07/30/2010  02:11 PM    <DIR>          SwSetup
01/31/2011  11:31 AM    <DIR>          temp
02/24/2012  02:40 PM          784,896 tsk-xview.exe
12/14/2010  02:58 PM    <DIR>          Users
10/01/2011  11:27 PM    <DIR>          Windows
    5 File(s)      842,338 bytes

```

14 Dir(s) 199,031,443,456 bytes free

```
C:\sandbox\hiddendirectory>cd sandbox
```

```
C:\sandbox\hiddendirectory\sandbox>dir
```

Volume in drive C has no label.

Volume Serial Number is 98B1-9C5A

Directory of C:\sandbox\hiddendirectory\sandbox

```

03/04/2012  05:39 PM    <DIR>          .
03/04/2012  05:39 PM    <DIR>          ..
07/13/2009  08:14 PM          301,568 cmd.exe
02/28/2012  03:21 PM              7 mike.txt
    2 File(s)      301,575 bytes
    2 Dir(s)      199,031,443,456 bytes free

```

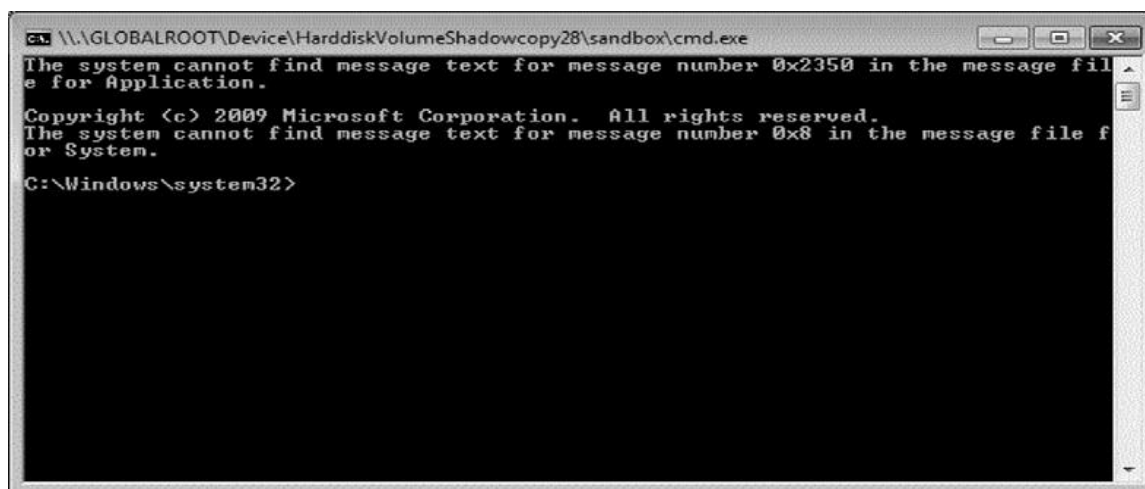
از آنجایی که در این مثال هدف بررسی ما وجود فایل اجرایی پنهان شده در Volume Shadow Copy و بررسی قابل اجرا بودن آن بوده، پس از آن می توانیم پیوند نمادین ایجاد شده را به روش زیر حذف کنیم.

```
C:\sandbox>rmdir hiddendirectory
```

تا این مرحله، بررسی کردیم که فایل اجرایی Cmd.exe قابل دسترسی و اجرایی است، بدون این که به وسیله ی فایل سیستم قابل مشاهده باشد و این کار را با استفاده از امکانات موجود در WMIC انجام دادیم. به این نکته مهم توجه کنید که در هنگام استفاده از Wmic از . به جای ؟ استفاده کنید.

```
C:\sandbox>wmic process call create
\\.\GLOBALROOT\Device\HarddiskVolumeShadowco
py28\sandbox\cmd.exe
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
    ProcessId = 5780;
    ReturnValue = 0;
};
```

اجرای فرمان Wmic باعث اجرای موفقیت آمیز فایل اجرایی پنهان مخفی Cmd.exe شده و پنجره ی مربوطه مانند شکل ۷-۱۱ باز می شود.



شکل ۷-۱۱: پنجره ی حاصل از اجرای فایل پنهان شده ی Cmd.exe در VSCS



با کنار هم گذاشتن تمام این امکانات، مشاهده می شود که از Volume Shadow Copy می توان برای پنهان کردن فایل ها استفاده کرد. اگر فایل پنهان شده اجرایی بود از WMIC می توان برای اجرای آن بدون نیاز به پیوند نمادین هم استفاده کرد؛ به علاوه بیشتر ویروس کش ها و پاد بدافزارها<sup>۱</sup> محتویات Volume Shadow Copy را اسکن نمی کنند پس Volume Shadow Copy مکانی عالی برای پنهان کردن فایل ها، بدافزارها و سایر فایل ها است. به این نکته مهم توجه کنید که محتویات Volume Shadow Copy دائمی نیست و برای ایجاد فضای لازم برای پشتیبان گیری فایل جدید، محتویات پیشین فایل ها پاک می شوند ولی برای PC خانگی، این زمان می تواند ۶ ماه یا بیشتر هم باشد و زمان طولانی را به بدافزار برای برجای گذاشتن خرابی در سیستم می دهد.

## پنهان سازی داده ها در لینوکس

با گسترش کاربرد لینوکس و حرکت به سوی استفاده از نرم افزارهای متنی باز و در دسترس بودن و مقیاس پذیر بودن آن ها و استقبال از لینوکس در سرورها باعث گسترش مقبولیت آن از سوی جامعه کاربری هم شده است. در نتیجه می توان رد لینوکس را در بسیاری از رایانه های خانگی و لب تاپ ها و تعداد بسیاری از ابزارهای موبایل پیدا کرد و این امر موجب پیدایش امکانات نامحدود پنهان سازی داده ها هم شده است. با چند مثال، پنهان سازی داده ها در لینوکس را بررسی می کنیم.

## نیرنگی که می توان در نام فایل ها در لینوکس به کاربرد

چند روش پایه برای پنهان کردن فایل ها و پوشه ها در لینوکس وجود دارد. مثلاً گذاشتن یک نقطه در آغاز نام فایل که باعث پنهان شدن فایل در پوشه می شود و برای کاربران لینوکس کاملاً شناخته شده است. ولی این روش تنها برای دستور LS مفید است. دستور ls-all، فایل های پنهان شده با این روش را هم نمایش می دهد.

<sup>۱</sup> Anti malware

```

spihuntr@spihuntrubuntu:~/sandbox2$ vi .mike.txt
spihuntr@spihuntrubuntu:~/sandbox2$ ls
spihuntr@spihuntrubuntu:~/sandbox2$ ls -al
total 12
drwxr-xr-x  2 spihuntr spihuntr      4096 2012-06-01 00:10  .
drwxr-xr-x 44 spihuntr spihuntr      4096 2012-06-01 00:10  ..
-rw-r--r--  1 spihuntr spihuntr      15 2012-06-01 00:10  .mike.txt

```

لیست گیری از پوشه، وجود ۰ و ۰۰ را هم آشکار می کند. نمایانگر پوشه جاری و ۰۰ نمایانگر پوشه ی پدر پوشه ی جاری است.

```

spihuntr@spihuntrubuntu:~/sandbox2$ mkdir  " . "
spihuntr@spihuntrubuntu:~/sandbox2$ ls -al
total 16
drwxr-xr-x  3 spihuntr spihuntr 4096 2012-06-01 00:11  .
drwxr-xr-x  2 spihuntr spihuntr 4096 2012-06-01 00:11  .
drwxr-xr-x 44 spihuntr spihuntr 4096 2012-06-01 00:10  ..
-rw-r--r--  1 spihuntr spihuntr 15 2012-06-01 00:10  .mike.txt

```

نکته ی جالب در این خصوص، امکان اضافه کردن Space بین این نقاط و ایجاد پوشه ی جدید است. بیشتر کاربران احتمالاً با نگاه گذرا به لیست از کنار تکرار ۰ به سادگی بگذرند. از این نکته می توان برای ایجاد پوشه ی پنهان دوم تحت نام "۰ ۰" استفاده کرد.

```

spihuntr@spihuntrubuntu:~/sandbox2$ mkdir  ".. "
spihuntr@spihuntrubuntu:~/sandbox2$ ls -al
total 20
drwxr-xr-x  4 spihuntr spihuntr 4096 2012-06-01 00:11  .
drwxr-xr-x  2 spihuntr spihuntr 4096 2012-06-01 00:11  .
drwxr-xr-x 44 spihuntr spihuntr 4096 2012-06-01 00:10  ..
drwxr-xr-x  2 spihuntr spihuntr 4096 2012-06-01 00:11  ..
-rw-r--r--  1 spihuntr spihuntr 15 2012-06-01 00:10  .mike.txt

```

این پوشه ها هیچ نشانه ای که مشخص کننده ی وجود نویسه Space باشد از خود بروز نمی دهند. بنابراین بسیاری از کاربران عادی و حرفه ای از کنار آن بدون توجه به این که این ها پوشه های ممکن است محتوا فایل ها پنهان باشند، می گذرند. این روش، راه حل ساده ای برای پنهان کردن فایل ها بوده و در بسیاری از نسخه های لینوکس ، مثل Mac os و Ubuntu کار می کند.

## پنهان سازی در سیستم فایل توسعه یافته

سیستم فایل توسعه یافته ext2، ext3، ext4 را در بسیاری از توزیع های لینوکس از Ubuntu گرفته تا Mac OS و اندرید می توان یافت. به علاوه این سیستم فایل مثل سایر فایل سیستم ها لینوکس و یونیکس حاوی inodes بوده و هر فایل و پوشه با یک inode مشخص می شود. هر inode حاوی اطلاعاتی در خصوص نوع فایل، حق دسترسی به فایل، نام صاحب فایل، برچسب های اندازه و اشاره گرها به بلوک داده های فایل است.

در سیستم فایل توسعه یافته وقتی فایلی حذف می شود، فایل، نام آن را به inode منتقل می کند ولی بخش داده های فایل تا زمانی که توسط سیستم فایل برای ایجاد فضای خالی برای فایل دیگری رونویسی overwrite نشود، حذف نمی گردد. وقتی فایلی حذف می شود و تا زمانی که بخش داده فایل توسط سیستم عامل بازنویسی نشده inodes تمام اطلاعات پیرامون فایل را نگهداری می کند. بنابراین داده ها را به روش های گوناگونی می توان بازیابی کرد ولی این روش ها به طور گسترده ای به ساختار سیستم فایل جاری بستگی دارند. شانس بازیابی فایل های حذف شده در سرور با حجم کار بالا و شلوغ خیلی کمتر از شانس بازیابی فایل در هارد اکسترنال بندانگشتی خانگی است.

با به کارگیری این روش در به سیستم فایل توسعه یافته و inodes می توانیم برای پنهان سازی فایل ها و بازیابی آن ها از inodes استفاده کنیم. لینوکس ابزاری برای بازیابی فایل های بی نام موجود در با inodes خود دارد ابزار بازیابی debugs در بیشتر توزیع های لینوکس وجود دارد اما کار با آن مشکل است. آقای اولیورد ریچ ابزار ساده تری به نام e2undel ساخته است.

کارمان را نخست با ایجاد سیستم فایل توسعه یافته برای حافظه اکسترنال شروع کرده و سپس راهی برای پنهان کردن داده ها در آن پیدا می کنیم. در این مثال از سیستم عامل ubuntu استفاده می کنیم. ابزارهای پارتیشن بندی گوناگونی در این سیستم عامل وجود دارد اما، ما از GParted استفاده می کنیم.

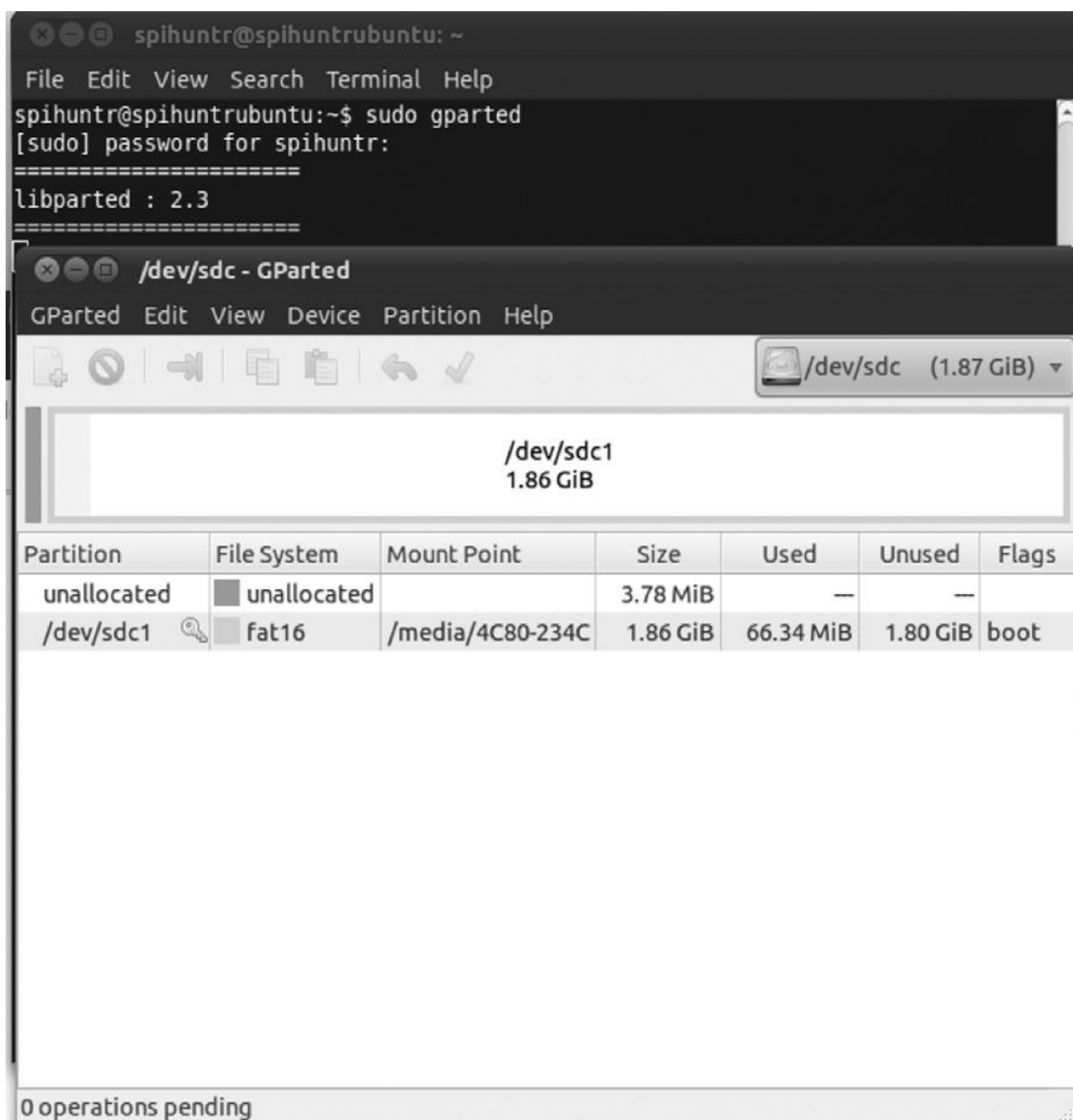
اگر در نسخه ی مورد استفاده ی شما این نرم افزار وجود ندارد، می توانید از آدرس زیر آن را دانلود

نمایید:

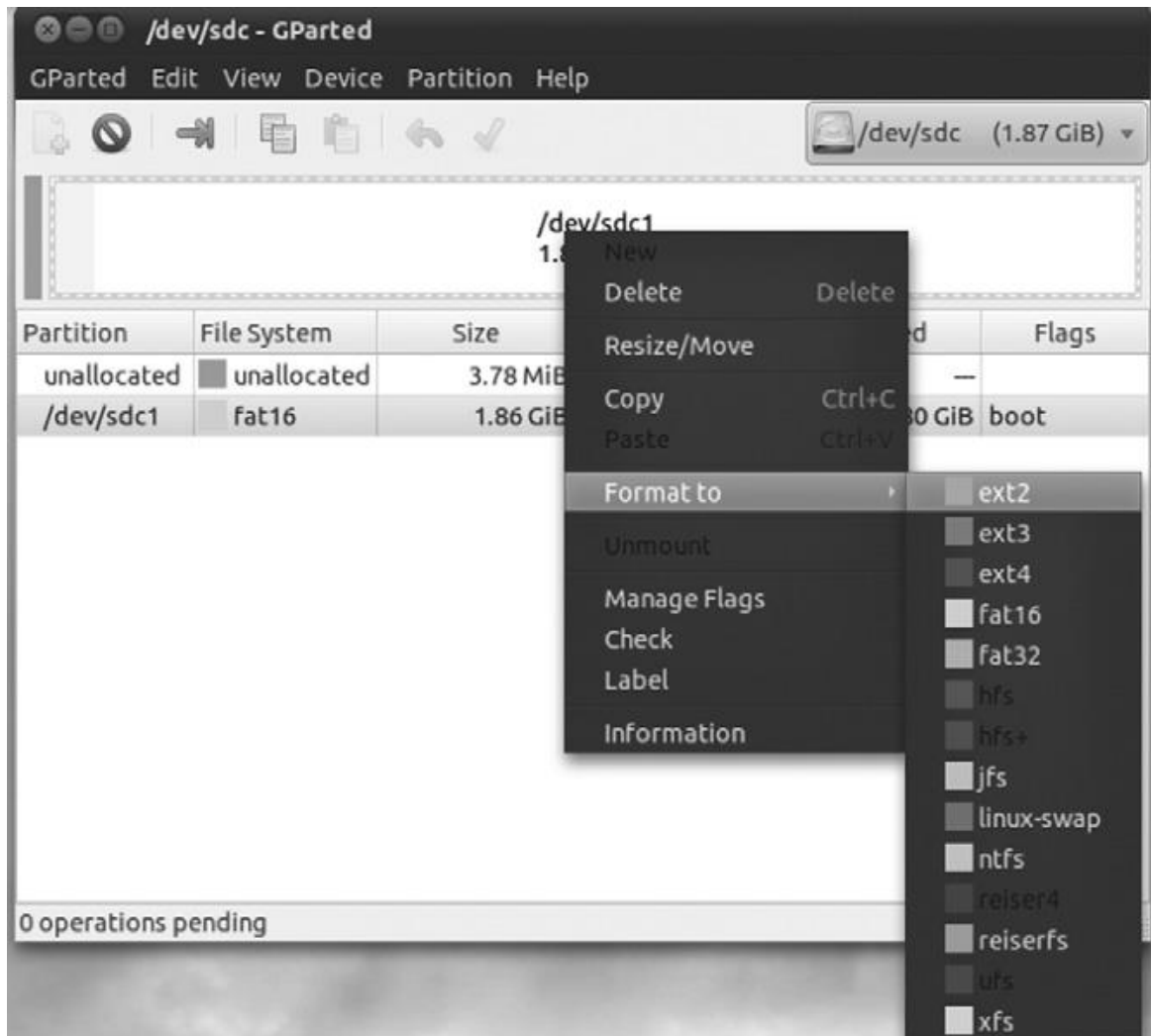
<http://gparted.sourceforge.net>

برای آغاز به کار نرم افزار از خط فرمان، مثل شکل ۷-۱۲ دستور gparted را اجرا کنید، سپس از لیست سمت راست، حافظه اکسترنالی که می خواهید فرمت نمایید را انتخاب کنید. پیش از فرمت، نسبت به انتخاب درست درایو مورد نظر خود مطمئن شوید تا برحسب اتفاق درایوهای مهم مثل درایو سیستم عامل را فرمت ننمایید. اگر درایو پیش تر بارگذاری شده بود، به سادگی و با استفاده از گزینه unmount

در درایو آماده فرمت نمایید گزینه ext2 را برای تعیین گونه‌ی توسعه یافته مثل شکل ۷-۱۳ انتخاب کنید.

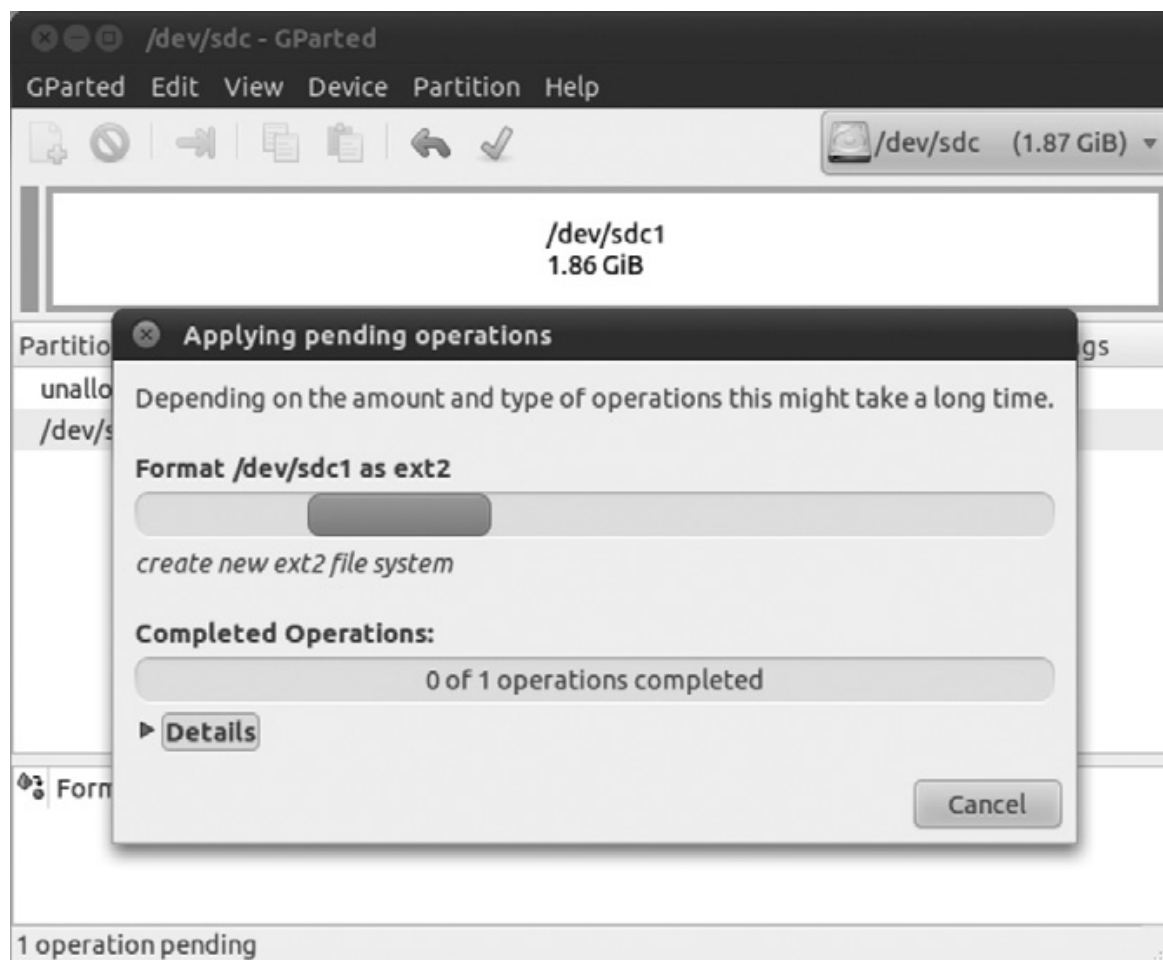


شکل ۷-۱۲: ایجاد پارتیشن در حافظه اکسترنال به وسیله‌ی Gparted



شکل ۷-۱۳: فرمت حافظه اکسترنال به شکل ext2

فرآیند فرمت را همانند شکل ۷-۱۴ آغاز نمایید. زمانی که فرآیند فرمت کامل شد پارتیشن ایجاد شده با برچسب ext2 همانند شکل ۷-۱۵ نمایش داده می‌شود.



شکل ۷-۱۴: حافظه اکسترنال در حال فرمت شده

| Partition   | File System | Size     | Used      | Unused   | Flags |
|-------------|-------------|----------|-----------|----------|-------|
| unallocated | unallocated | 3.78 MiB | —         | —        |       |
| /dev/sdc1   | ext2        | 1.86 GiB | 32.80 MiB | 1.83 GiB | boot  |

شکل ۷-۱۵: حافظه ی اکسترنال فرمت شده به شکل ext2

~~~~~

حالا که سیستم فایل توسعه یافته از گونه ی ext2 را در اختیار داریم، شروع به پنهان سازی داده ها در آن می کنیم. روش کار ما کپی از فایل های موجود و ایجاد فایل جدید بر روی حافظه اکسترنال و حذف اصل فایل ها می باشد.

اگر خواستید فایل های حذف شده را دوباره بازیابی کنید از ابزار e2undel به شکل دستور زیر استفاده نمایید:

```
e2undel -d device -s path [-a] [-t]
with
-d device: the file system where to look for deleted files (like /dev/
hda1)
```

D : مشخص کننده ی نام درایوی است که می خواهید فایل های حذف شده را از آن بازیابی نمایید .

S : مسیر بازیابی فایل را تعیین می کند .

A: لیست log فایل های حذف شده را نادیده گرفته و تمام فایل های حذف شده را بازیابی می کند (زمانی استفاده از گزینه مناسب است که بخواهید فایل حذف شده ای را بازیابی نمایید که پیش از نصب این ابزار حذف شده است.

T: سعی در بازیابی فایل ها براساس نوع آن ها بدون در نظر گرفتن نام فایل دارد و تنها با -a کار می کند.

اکنون ما دو فایلی را که پیشتر حذف کردیم بازیابی می کنیم. نام درایو و مسیری که می - خواهیم فایل های بازیابی شده را ذخیره کنیم به شکل زیر تعیین می کنیم:

```
spihuntr@spihuntrubuntu:~$ sudo e2undel -d /dev/sdc1 -s
/home/spihuntr/sandbox -a -t
```

```
e2undel 0.82
```

```
Trying to recover files on /dev/sdc1, saving them on
/home/spihuntr/sandbox
```

```
/dev/sdc1 opened for read-only access
```

```
/dev/sdc1 was not cleanly unmounted.
```

```
Do you want to continue (y/n)? y
```

```
122160 inodes (122149 free)
```

```
487992 blocks of 4096 bytes (479595 free)
```

```
last mounted on Wed Dec 31 19:00:00 1969
```

```
/dev/sdc1 is mounted. Do you want to continue (y/n)? y
```

```
reading log file: opening log file: No such file or directory
```

```
no entries for /dev/sdc1 in log file
```

```
searching for deleted inodes on /dev/sdc1:
```

```
|=====|
```

```
122160 inodes scanned, 2 deleted files found
```

| user name | 1 <12 h | 2 <48 h | 3 <7 d | 4 <30 d | 5 <1 y | 6 older |
|-----------|---------|---------|--------|---------|--------|---------|
| spihuntr  | 2       | 0       | 0      | 0       | 0      | 0       |

نخست نام کاربری پرسیده شده و همان گونه که پیشتر گفتم از اطلاعاتی که توسط inode در

خصوص فایل های حذف شده برای ساختن جدولی از فایل های حذف شده استفاده می کنیم . در پنجره ی

بعدی بازه ی زمانی که در آن فایل های حذف شده را می خواهیم بازیابی کنیم را مشخص می نماییم.



```

Select an inode listed above or press enter to go back: 12
15 bytes written to /home/spihuntr/sandbox/inode-12-ASCII_text
Select an inode listed above or press enter to go back: 13
27 bytes written to /home/spihuntr/sandbox/inode-13-data
Select an inode listed above or press enter to go back:
  user name | 1 <12 h | 2 <48 h | 3 <7 d | 4 <30 d | 5 <1 y | 6 older
-----+-----+-----+-----+-----+-----+-----+
  spihuntr  |      2 |      0 |      0 |      0 |      0 |      0
Select user name from table or press enter to exit:
spihuntr@spihuntrubuntu:~$

```

اکنون جزئیات فایل هایی که می خواهیم بازیابی کنیم را نمایش می دهد. اگر بخواهیم که دو فایل حذف شده در ردیف ۱۲ و ۱۳ را بازیابی نماییم به شکل زیر عمل می کنیم:

```

Select an inode listed above or press enter to go back: 12
15 bytes written to /home/spihuntr/sandbox/inode-12-ASCII_text
Select an inode listed above or press enter to go back: 13
27 bytes written to /home/spihuntr/sandbox/inode-13-data
Select an inode listed above or press enter to go back:
  user name | 1 <12 h | 2 <48 h | 3 <7 d | 4 <30 d | 5 <1 y | 6 older
-----+-----+-----+-----+-----+-----+
  spihuntr  |      2 |      0 |      0 |      0 |      0 |      0
Select user name from table or press enter to exit:
spihuntr@spihuntrubuntu:~$

```

فایل های بازیابی شده باید در پوشه ی مقصد تعیین شده در دستور بالا ذخیره شده باشد. با مشاهده ی این پوشه می توانیم فایل ها را ملاحظه نمایید. از آنجایی که اسامی فایل ها در زمان حذف فایل پاک می شوند به فایل های بازیابی شده به طور خودکار نام اختصاص داده می شود.

```

spihuntr@spihuntrubuntu:~/sandbox$ ls -al
total 32
drwxr-xr-x 2 spihuntr spihuntr 4096 2012-05-30 22:21 .
drwxr-xr-x 2 spihuntr spihuntr 4096 2012-05-30 22:23 ..
-rwxr-xr-x 1 root      root      15   2012-05-31 14:33 inode-12-
ASCII_text
-rwxr-xr-x 1 root      root      27   2012-05-31 14:33 inode-13-data

```

اگر مروری بر کارمان داشته باشیم، برای پنهان کردن فایل ها، نخست آن ها را حذف کرده سپس با ابزار

“e2undel” آن ها را بازیابی نمودیم . اگر محتویات فایل را ملاحظه کنید مشاهده می شود داده های فایل اصلی را در خود دارد .

```
spihuntr@spihuntrubuntu:~/sandbox$ more inode-12-ASCII_text
hidden message
spihuntr@spihuntrubuntu:~/sandbox$
```

روش به کاررفته رفته موفق بود و حتی محدود به استفاده در سیستم فایل توسعه یافته ی ext2 هم نیست. به عنوان مثال با استفاده از نرم افزار debugfs می توان فایل هایی را از گونه ext3 و ext4 را هم بازیابی کرد. به علاوه اگرچه ما این روش را در توزیع سیستم عامل uhn... به کار بردیم، ولی بر روی Mac OS, Red Hat, Android و سایر توزیع های سیستم فایل توسعه یافته نیز به خوبی کار می کند. اما توجه داشته باشید که این روش بی نقص نیست. بسیاری از سیستم های فایل های فعال معمولاً بر روی داده های فایل های حذف شده رونویسی می کنند و این باعث کوتاه شدن عمر فایل های حذف شده برای بازیابی آن ها شده است.

اما در لپ تاب شخصی که فضای آزاد بیشتری برای ذخیره سازی وجود دارد این روش مفیدتر و کارا تر عمل می کند.

این نرم افزار امکان پیاده سازی و اجرای رمزنگاری همزمان با دسترسی on-the-fly-encrypted volume را در ابزارهای ذخیره سازی داده ها می دهد .

Onthe-fly encryption امکان رمزنگاری داده ها را درست پیش از ذخیره سازی آن ها و رمزگشایی آن ها را پس از بارگذاری در حافظه و بدون دخالت کاربر می دهد. هیچ داده ی رمزنگاری شده در هارد را نمی توان رمزگشایی کرد، مگر کلید یا گذر واژه رمزگشایی را در اختیار داشته باشید. کل سیستم فایل رمزنگاری می شود، از جمله نام فایل، نام پوشه، محتویات تک تک فایل ها، فضای آزاد هارد، ابر داده ها و ... .

همچنین نرم افزار TrueCrypt مفهوم هارد volume را ارائه می کند . برای برخی این به معنی پناهگاه قابل اعتماد در زمان رویارویی با دشمن است. این پناهگاه موفقیت یا مکانی است که اسناد اندک یا هیچ سندی دال بر سوء استفاده از آن وجود ندارد.

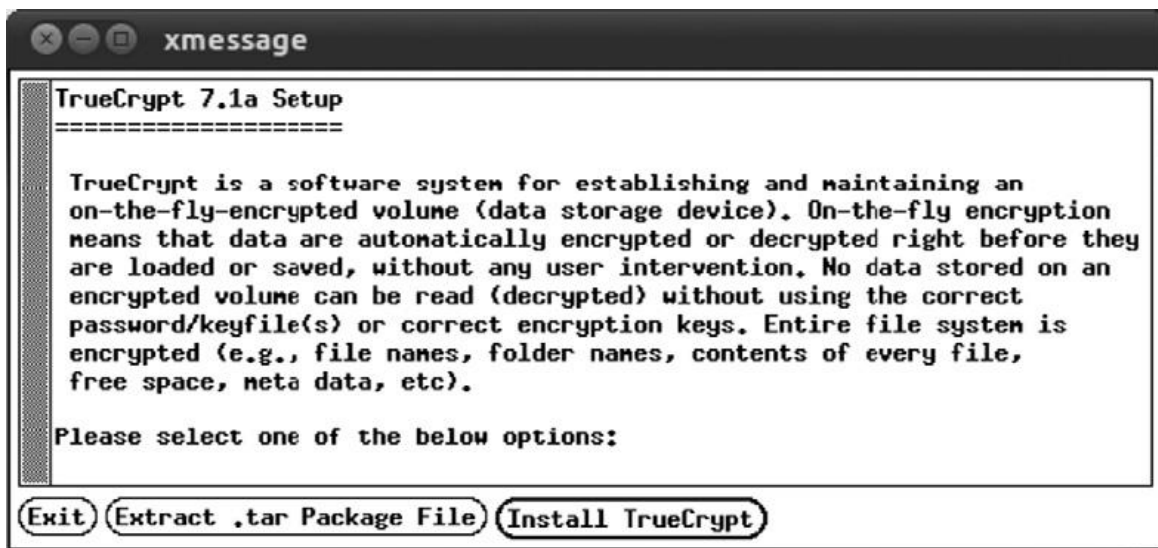
به زبان قانونی، به مکانی اطلاق می شود که مدارک ناچیزی برای اثبات اتهام در خصوص آن وجود دارد.

پیدا کردن راهی که به کاربران اجازه می دهد با استفاده از این پناهگاه به طور محسوس مانع از انتشار اسنادی مهم برای رقیب شوند بسیار مهم است.

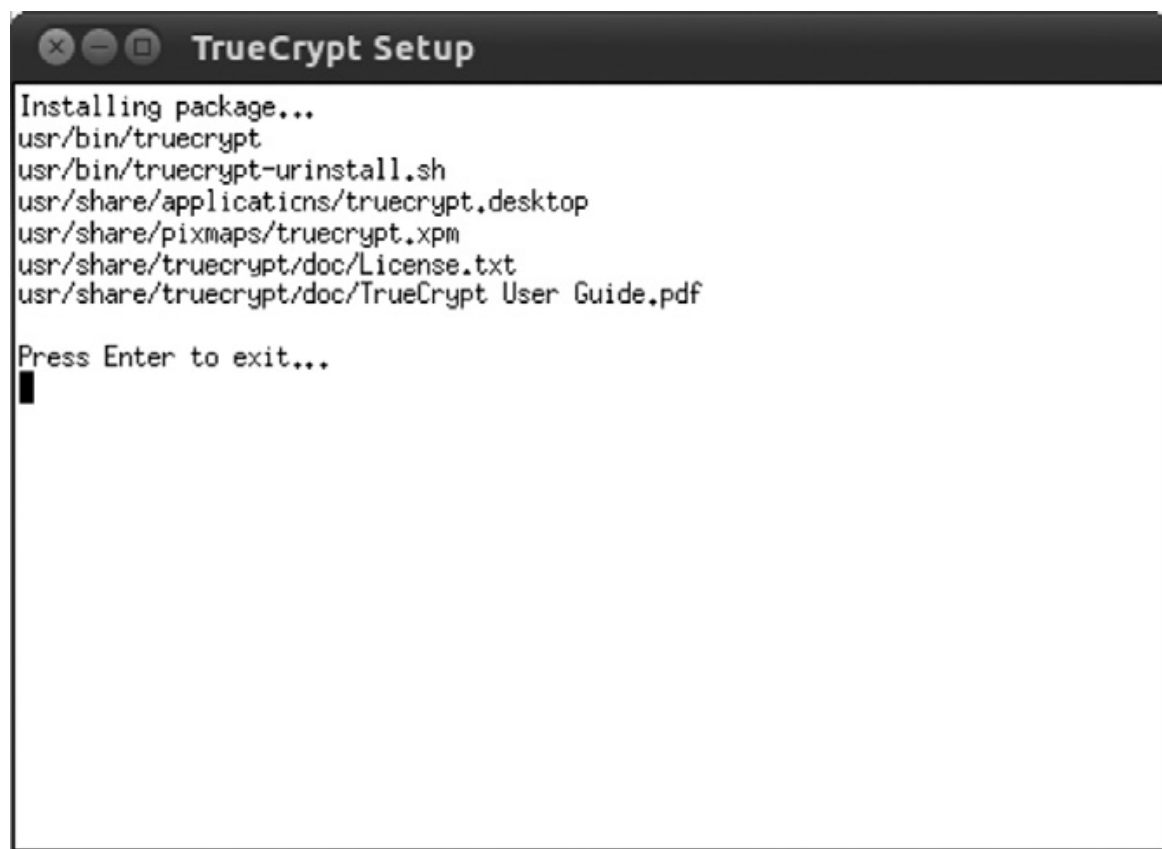
طراحی TrueCrypt به گونه ای است که تحلیل ها هیچ هدر یا داده ی شناخته شده ای را مبنی بر به کارگیری آن آشکار ننموده اند و داده ها بیشتر تصادفی محض بیت ها به نظر می رسند. پرونده ی کاربرد این نرم افزار و پناهگاه مربوطه در دنیای واقعی، استفاده از آن توسط بانک های مظنون از دید دولت برزیل به دست داشتن در جرائم مالی بود. پلیس برزیل ۵ هارد دیسک که با استفاده از TrueCrypt محافظت می شد را توقیف کرده و پس از ماه ها تحلیل و بررسی بی نتیجه، انستیتو ملی جرم شناسی برزیل ناچار شد ادامه تحقیقات را به FBI بسپرد اما پس از سپری شدن ۱۲ ماه FBI هم در شکستن رمز آن ها ناکام ماند.

TrueCrypt توجه جهانی را به خود جلب کرد و ارزش بررسی ژرف تر چگونگی عملکردش را پیدا کرد.

اجازه دهید بررسی خود را با نصب و استفاده از این نرم افزار شروع کنیم (عکس ۷-۱۶ و ۷-۱۷).



شکل ۷-۱۶: صفحه شروع نصب نرم افزار true crypt



شکل ۷-۱۷: مراحل نصب package نرم افزار

TrueCrypt بر روی سیستم عامل های گوناگونی چون ویندوز لینوکس اجرا می شود. نسخه ای از آن را می توانید سایت [truecrypt.org](http://truecrypt.org) دریافت نمایید.

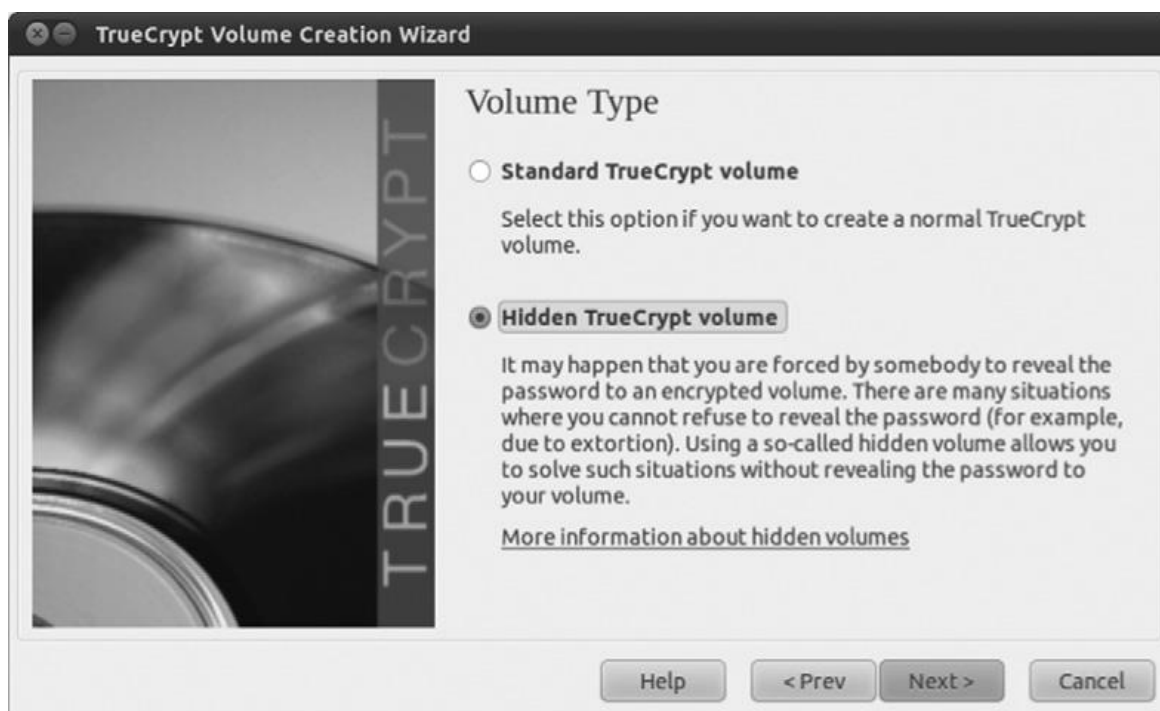
پس از نصب می توان با دستور زیر از خط فرمان لینوکس نرم افزار را اجرا کرد.

```
spihuntr@spihuntrubuntu:~$ truecrypt
```

در این بررسی از حافظه ی فلش ۲GB برای ایجاد درایو TrueCrypt حاوی volume پنهان استفاده می کنیم. بنابراین پس از اتصال فلش به لپ تاب گزینه Create a volume with a partition/drive را برای ایجاد پارتیشن پنهان همانند شکل ۷-۱۹ انتخاب می کنیم.



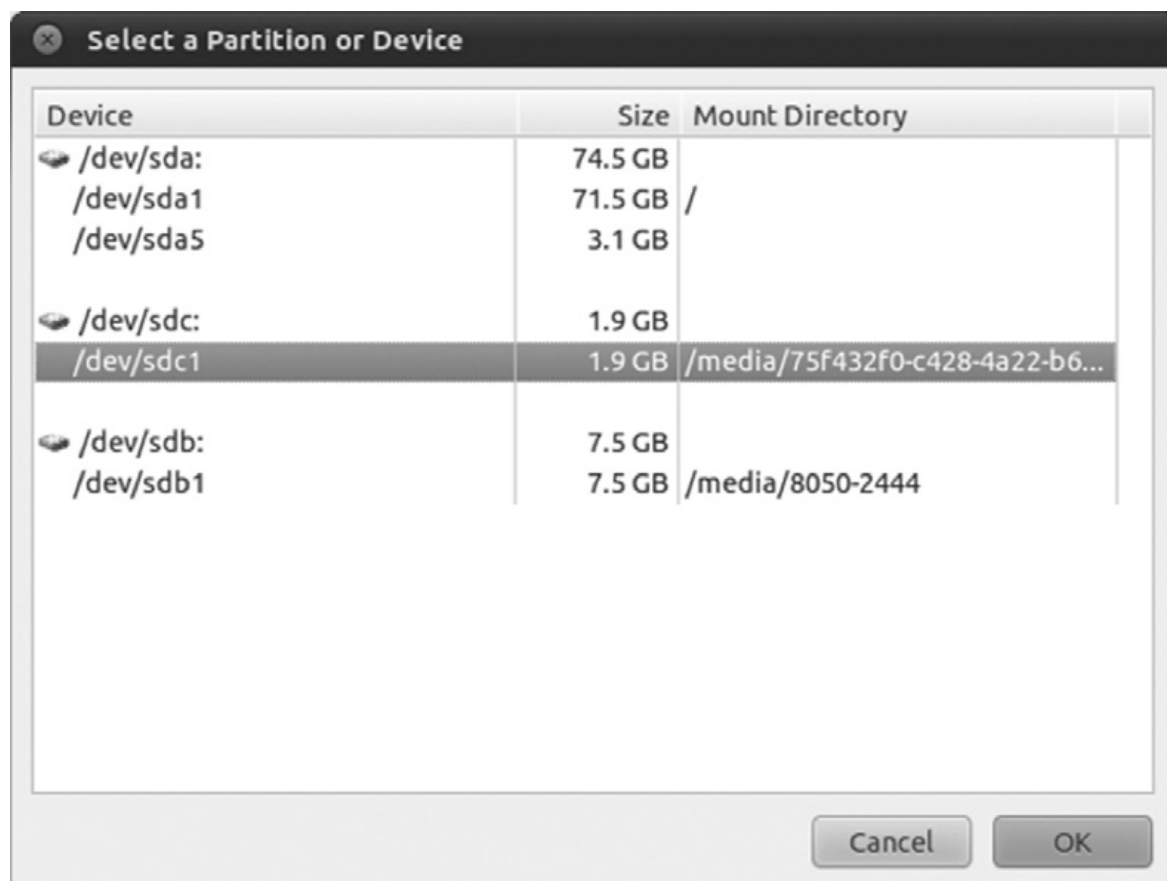
شکل ۷-۱۸: چگونگی ایجاد پارتیشن یا درایو پنهان



شکل ۷-۱۹: مشخص کردن درایو پنهان

در مرحله ی بعدی باید حافظه یا پارتیشن مورد نظر و چگونگی رمزنگاری را همانند عکس های ۷-۲۰ تا

۷-۲۲ مشخص می کرد.



شکل ۷-۲۰: انتخاب درایو مورد نظر برای ایجاد پارتیشن پنهان



شکل ۷-۲۱: انتخاب شیوه رمزنگاری AES برای رمزنگاری پارتیشن



شکل ۷-۲۲: انتخاب گذرواژه برای دسترسی به پارتیشن میزبان

با این کار میزبان یک یا چند پارتیشن را همانند شکل ۷-۲۳ تعیین می‌نماییم .



شکل ۷-۲۳: فرمت پارتیشن میزبان

پس از اجرای مراحل بالا، به سادگی می توان راه ورود خود را به درایو پنهان با اجرای truecrypt در خط فرمان لینوکس همانند شکل های ۷-۲۴ تا ۷-۲۷ هموار نمود.



شکل ۷-۲۴: ویزارد ایجاد volume در پارتیشن پنهان



شکل ۷-۲۵: گزینه payload رمزنگاری volume پنهان



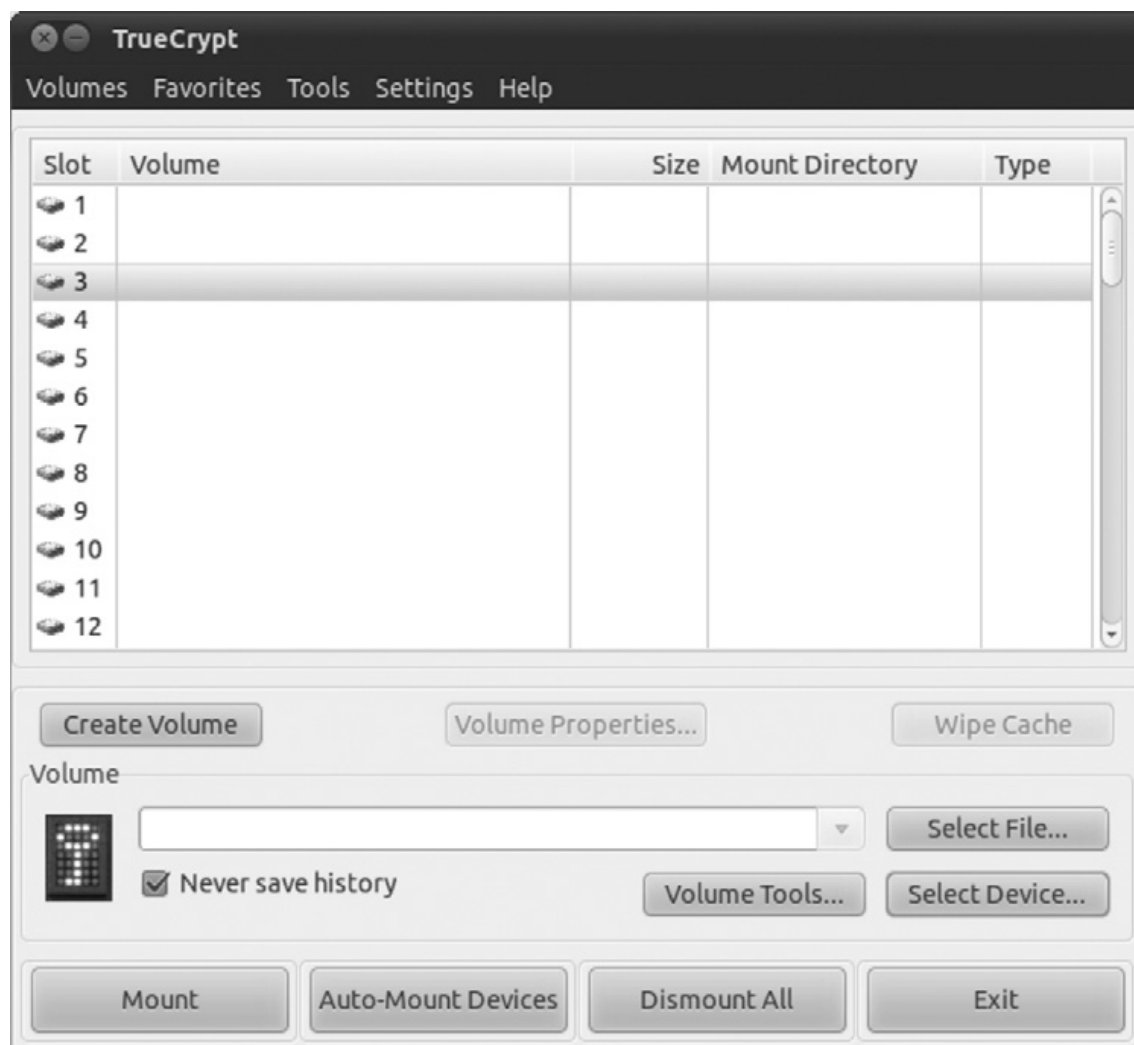


شکل ۷-۲۶: گزینه تعیین اندازه volume پنهان



شکل ۷-۲۷: payluad نوع سیستم فایل

پس از پایان مراحل بالا ، می توانیم پارتیشن را بارگذاری و از آن استفاده نماییم. برای این کار گزینه Select Device را برای دسترسی به فلش GB۲، همانند شکل ۷-۲۸ انتخاب نماییم.



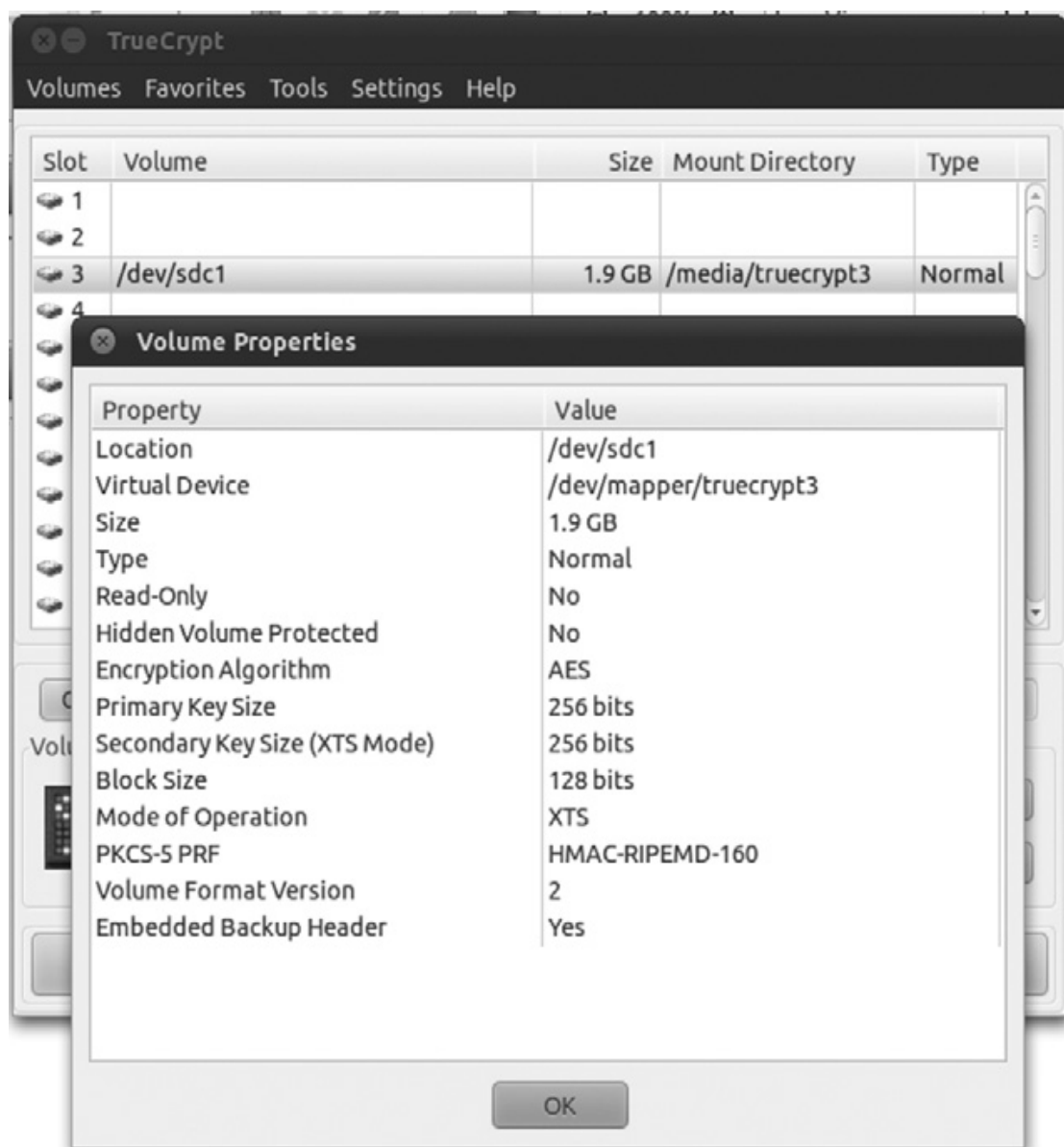
شکل ۷-۲۸: چگونگی بارگذاری و دسترسی به درایو تازه ایجاد شده

برای دسترسی به این درایو، گذرواژه‌ای را که در حین فرآیند نصب برگزیدید وارد کنید تا درایو بارگذاری و اجازه‌ی دسترسی به آن به شما داده شود (عکس ۷-۲۹).



شکل ۷-۲۹: تایپ گذرواژه تعیین شده در مراحل نصب برای دسترسی به درایو .

گزینه properties اجازه‌ی مشاهده‌ی جزئیاتی درباره‌ی TrueCrypt و چگونگی رمزگذاری درایو را می‌دهد (عکس ۷-۳۰ و ۷-۳۱).



شکل ۷-۳۰: مشاهده جزئیات درایو TrueCrypt



شکل ۷-۳۱: دسترسی به فایل‌ها در درایو ایجاد شده به وسیله‌ی true crypt

حال درایو تا زمان unmounted در دسترس شما است. همان‌گونه که اشاره شد نصب TrueCrypt ساده و تنها نیاز به اجرای ویزارد دارد.

این سطح از رمزنگاری، داده را تصادفی جلوه داده و پتانسیل پر قدرتی در پنهان‌سازی داده‌ها در درایو دارد. این نرم‌افزار تحت ویندوز هم اجرا شده، در نتیجه قدرت بالقوه این برنامه را تقریباً در اختیار همگان است. در بازرسی قضایی، کشف استفاده از TrueCrypt برای رمزنگاری داده‌ها چالش بزرگی است و بازیابی داده‌ها بدون گذرواژه تا زمان انتشار این کتاب تقریباً غیرممکن است.

شایان توجه است پیتز کلینزدر Black Hat USA 2009 توضیح داد چگونه TrueCrypt's MBR<sup>۱</sup> می‌تواند کل رمزنگاری درایو را نادیده بگیرد.

ولی تنها در برخی شرایط کار می‌کند. این شرایط به ویژه زمانی است که کاربر از bootkit نامعتبر اجرا کرده و با سطح دسترسی کامل به سیستم وارد شده باشد و در حین دسترسی فیزیکی به درایو هم ممکن باشد. به عبارت دیگر اگر قصد استفاده از TrueCrypt را دارید از برترین تجارب امنیتی خود استفاده کنید.

<sup>۱</sup> Master Boot Record

~~~~~

## فصل هشتم

### پنهان سازی داده ها به شکل مجازی

بسیاری از شرکت ها، قوانین سخت گیرانه ای اعمال می کنند تا کاربران شان از تأثیر بدافزارها، ویروس ها، اسب تراواها در امان باشند و مانع استفاده های نامطلوب از رایانه هایشان شوند. با گسترش استفاده از ماشین های مجازی و محیط های مجازی، کاربران، راه های استفاده از ماشین های مجازی برای دسترسی به برنامه ها و صفحات اینترنتی را که توسط شرکتشان بلوک شده بود، را پیدا کرده اند. به علاوه کاربران بدخواه هم از ماشین های مجازی برای ناشناس ماندن هنگام سرقت اسرار شرکت یا سرقت داده های محرمانه مثل اطلاعات شخصی کارکنان یا اطلاعات کارت های اعتباری استفاده می کردند.

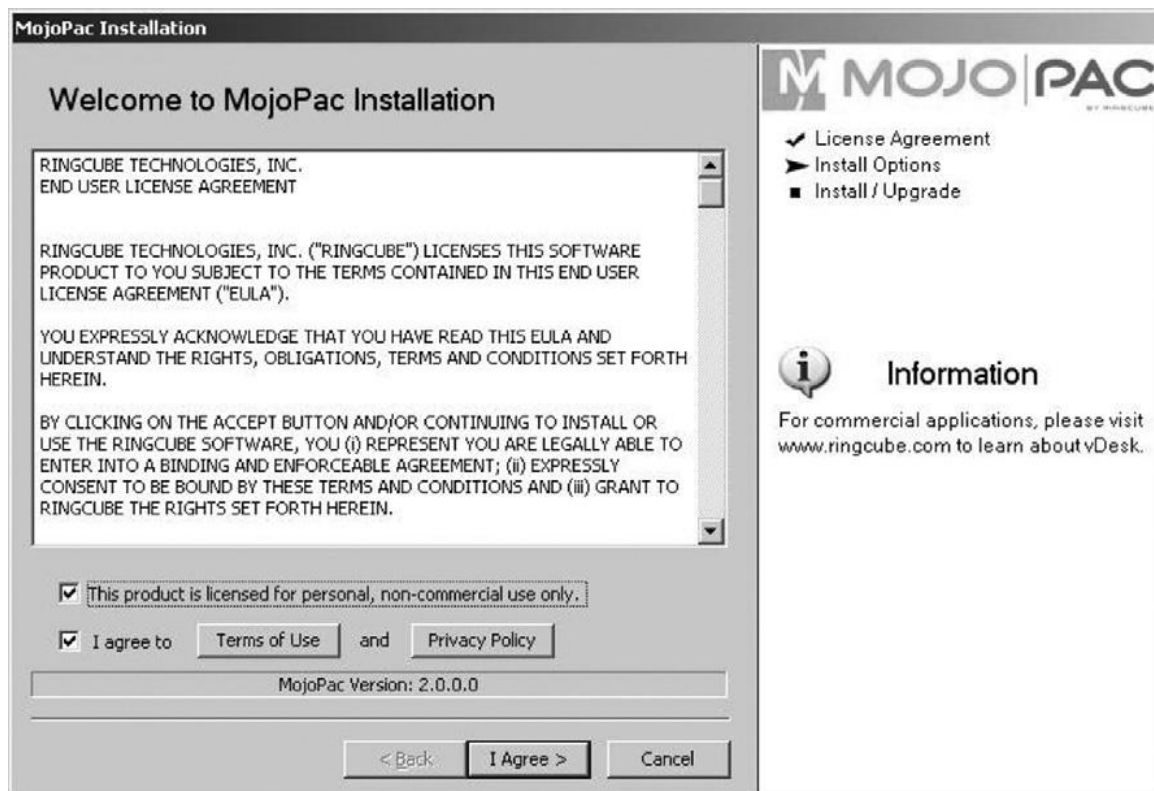
تشخیص این گونه محیط های مجازی در کل شبکه، چالش رو به گسترش مدیران شبکه است. این-گونه محیط های مجازی می توانند از کشف شدن به وسیله ویروس کش ها، پویشرهای شبکه و حفاظت end-point بگریزند، زیرا بسیاری از این گونه نرم افزارها، ماشین مجازی را پوشش نمی کنند. برای تشریح بیشتر مشکلات تشخیص آن ها به همین جا بسنده می کنیم که برخی از این ماشین های مجازی توانایی اجرا شدن از کارت های حافظه SD و حافظه های فلش USB را دارند. مثلاً نسخه ی قابل حمل Virtual box را می توان از روی حافظه های USB هم اجرا کرد. سایر محیط های مجازی به گونه ای طراحی شده-اند تا از رایانه ای به رایانه دیگر قابل حمل باشند، مثل Mojo-pac.

## پنهان سازی در محیط مجازی

نرم‌افزار Mojo-pac امکان اجرای نرم‌افزار مولد محیط مجازی از روی حافظه USB را داشته و می‌تواند از رایانه‌ای به رایانه دیگر هم منتقل شود، تا به شما اجازه‌ی اجرای محیط کوک شده Xp خاص خود را بر روی هر رایانه‌ای بدهد. بدون در نظر گرفتن کاستی‌هایش با داشتن مزیت قابل حمل بودن و سادگی کار با آن، این نرم‌افزار به انتخابی جذاب برای کاربرانی که قصد پنهان کردن داده‌ها در محیط مجازی بر پایه Desktop را دارند، تبدیل شده است.

## نصب نرم‌افزار

برای شروع نصب نرم‌افزار، نخست آخرین نسخه‌ی mojo-pac را دانلود کنید. یک عدد فلش USB ترجیحاً تازه و بدون هیچ فایلی، هم مورد نیاز است. Mojo-pac با این فلش همانند یک درایو استاندارد در ویندوز مثل درایو C:\ رفتار می‌کند (عکس ۸-۱).

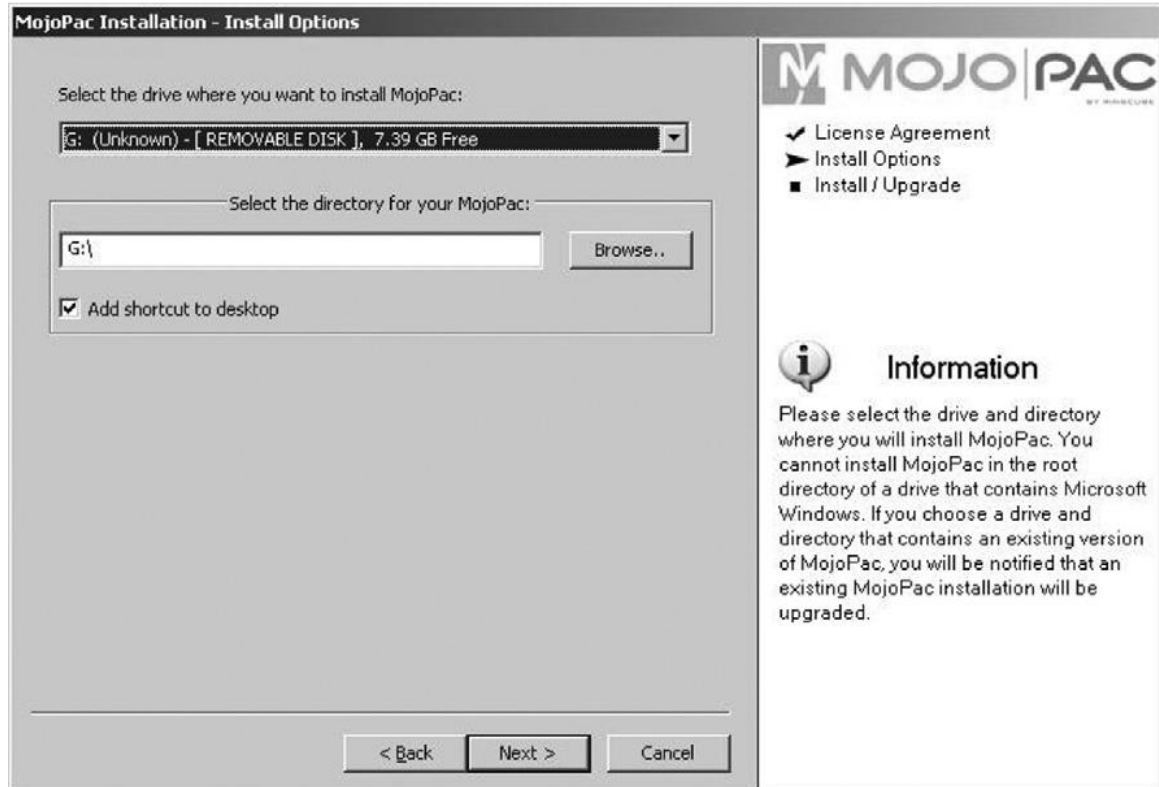


شکل ۸-۱: نصب نرم‌افزار Mojo pac

وقتی نصب را شروع کردید در پنجره‌ی دوم از شما خواسته می‌شود درایو مقصد را مشخص نمایید و شما باید فلش USB را انتخاب کنید! دقت کنید اگر می‌خواهید استفاده از این نرم‌افزار را پنهان نمایید

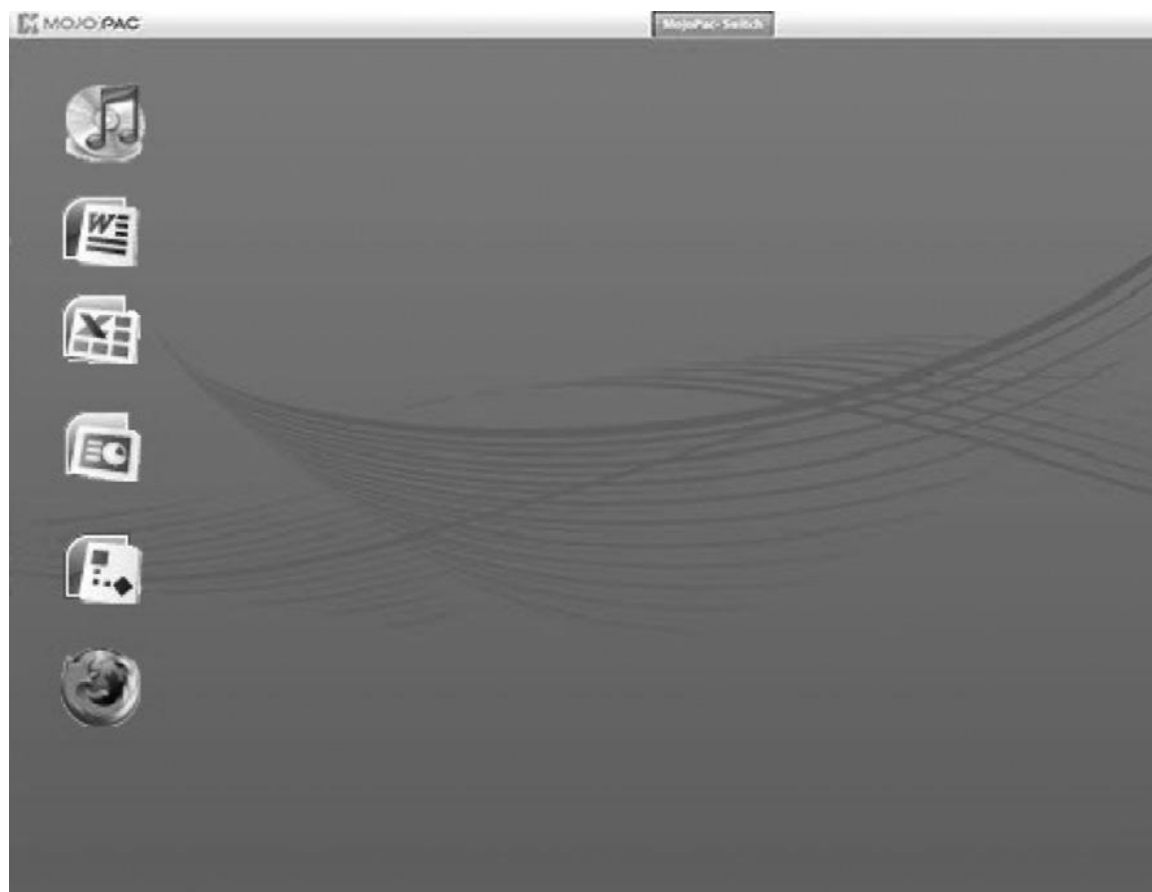


که ایده ی خوبی هم هست، باید گزینه ی ADD shortcut را پیش از ادامه ی کار غیرفعال کنید (عکس ۲-۸).



شکل ۲-۸: انتخاب درایو مقصد ماشین مجازی نرم افزار Mojo pac

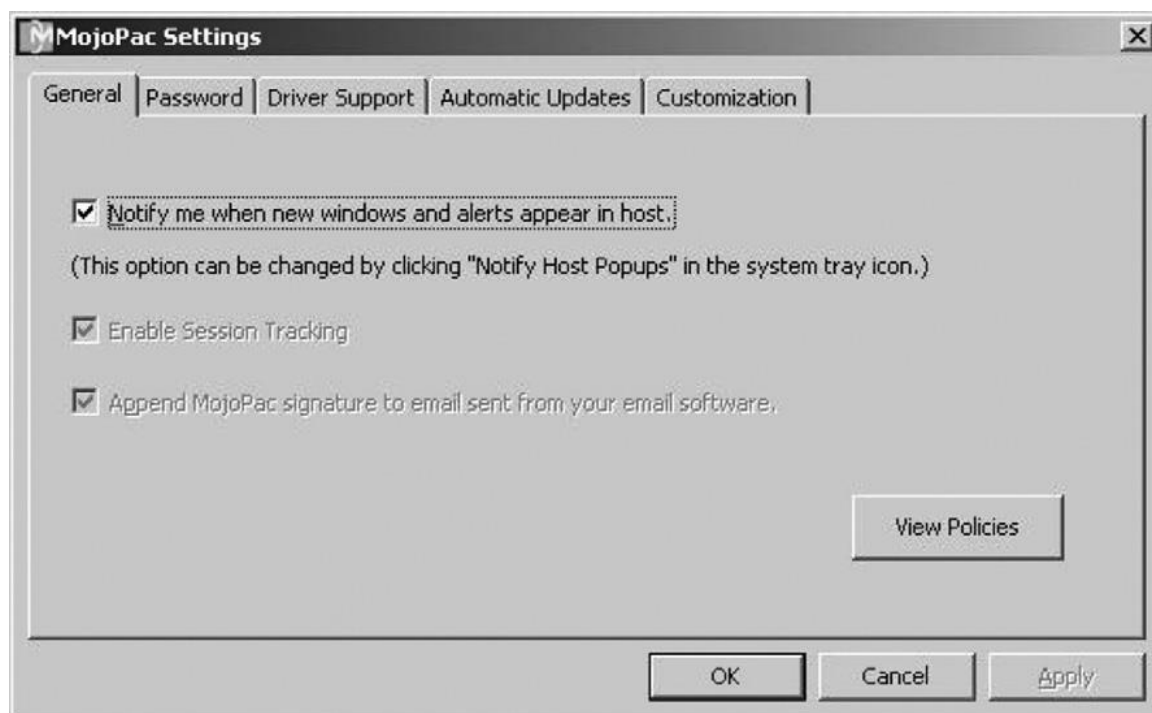
سپس برنامه شروع به آماده سازی درایو برای استفاده ی ماشین مجازی می کند. با ادامه ی مراحل نصب برنامه، صفحه Desktop آن اجرا می شود (عکس ۳-۸).



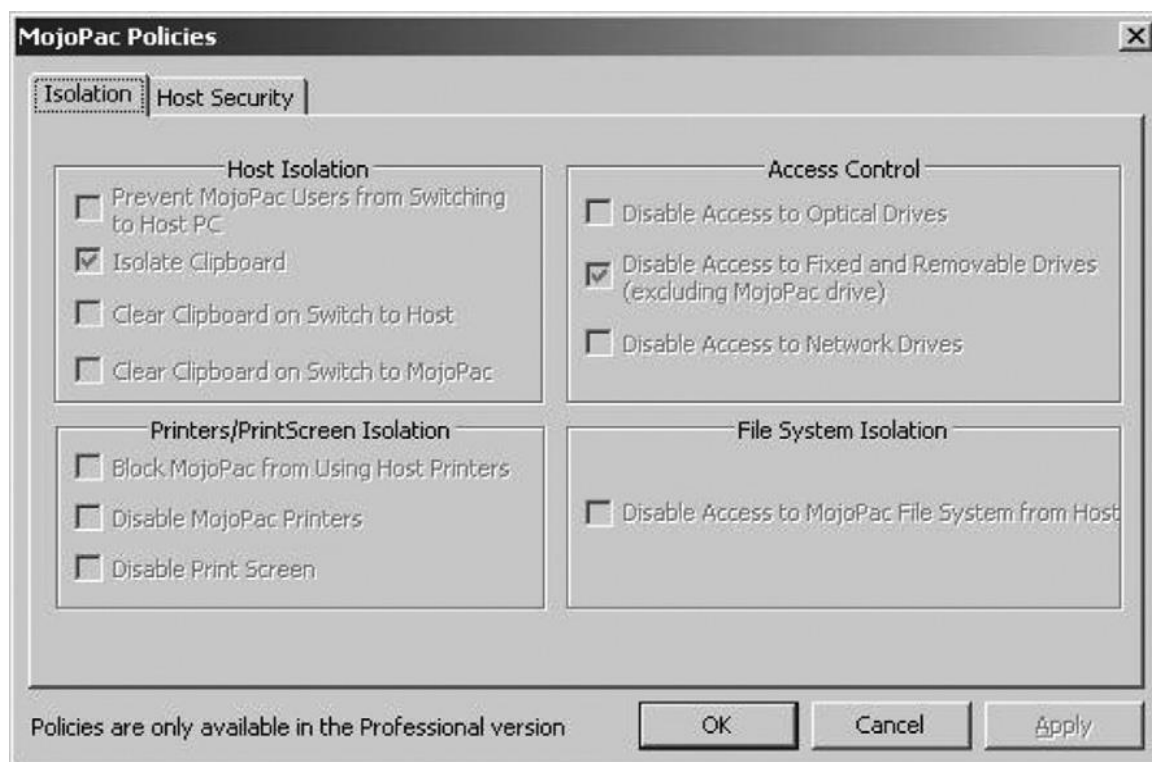
شکل ۸-۳: صفحه Desktop نرم افزار Mojo pac

حال نرم افزار Mojo Pac نصب شده در فلش USB را می توان از رایانه ای با سیستم عامل XP به رایانه دیگری با همان سیستم عامل منتقل کرد و بنابراین به کاربران اجازه می دهد از خطر کشف شدن بگریزند.

به این نکته توجه کنید که بقایای به جامانده از به کارگیری نرم افزار Mojo Pac در رایانه ای با سیستم عامل XP، چگونگی استفاده از آن را آشکار می کند. جزئیات اشاره شده در مقاله ی Virtualization and Forensics، نوشته ی Barrett بیانگر چگونگی استفاده مزنون از نرم افزار Mojo pac در رایانه را می توان در فایلی پیدا کرد. پس در سربرگ General با انتخاب گزینه ی Policies تعامل بین رایانه و محیط مجازی Mp را پیکربندی و یا محدود کنید (عکس ۸-۴ و ۸-۵).



شکل ۸-۴: پیکربندی نرم‌افزار mojo pac



شکل ۸-۵: پیکربندی سیاست‌های در Mojo pac

با این پیکربندی می‌توانید دسترسی به درایوهای داخلی و ابزارهای حافظه‌ی همراه را محدود کرده و دسترسی به سیستم فایل سیستم Mojo pac را به وسیله‌ی رایانه میزبان غیرفعال کنید. از این گزینه برای گریز از کشف شدن در شبکه یا به وسیله‌ی سیستم تشخیص مهاجم در شبکه IDS بر پایه میزبان استفاده می‌شود.

## مروری بر محیط‌های مجازی

پیش‌تر که نرم‌افزار Mojo Pac را بررسی کردیم تا دریافتیم که چگونه محیط مجازی را عنوان در داخل سیستم‌عامل رایانه میزبان ایجاد کرد. ولی ماشین‌های مجازی دیگری هم وجود دارد که فراگیرترند، مانند vmwave و این‌گونه ماشین‌های مجازی کل سیستم‌عامل را دربر می‌گیرند.

معمولاً اگر بخواهیم حجم زیادی از داده‌ها را پنهان کنیم، فایل‌های چندرسانه‌ای به لحاظ حجم بزرگشان گزینه‌ی مناسبی هستند. همان‌گونه که در فصل‌های پیشین بیان شد، پراکندن داده‌های پنهان در فایل چندرسانه‌ای با MB10 حجم، کمترین تأثیر را بر خود فایل چندرسانه‌ای دارند و در مقایسه با یک فایل عکس با MB2 حجم، امکان پنهان‌سازی مقدار بیشتری داده را فراهم می‌کند و حاصل آن، فایل چندرسانه‌ای است که کیفیت صوت و تصویر آن دگرگون‌چندانی نداشته است. اما اگر بخواهیم حجم زیادی داده مثال MB2 را داخل فایل عکس Jpeg پنهان کنیم، حاصل آن تصویری مبهم و تکه‌تکه است.

به لحاظ حجم بالا، ماشین مجازی مکانی عالی برای پنهان کردن مقدار زیادی داده است. ولی برخلاف ماهیت ایستا فایل چندرسانه‌ای، اندازه‌ی فایل ماشین مجازی، بسته به نوع استفاده، اندازه‌ی متغیری دارد. برای پیشی گرفتن از روش‌های کشف داده‌های پنهان و گریز از برنامه‌های شناخته شده، می‌بایست آزمون جامعیت را که تغییرات نامعمول فایل را تشخیص می‌دهد برای پنهان کردن داده‌ها در ماشین مجازی به کار ببریم؛ پس، نخست باید اجزاء تشکیل دهنده‌ی ماشین مجازی را درک کنیم.

## فایل‌های VMware

برای پنهان‌سازی داده‌ها در ماشین مجازی، می‌بایست نخست اجزاء ماشین مجازی را بشناسیم. Image ماشین مجازی VMware، معمولاً شامل گروه کوچکی از فایل‌هاست که نوع و ماهیتشان به شرح زیر است (عکس ۸-۶):

```

C:\WINDOWS\system32\cmd.exe
C:\Virtual Machines\Backup\Ubuntu 8.04 JeOS>dir
Volume in drive C is System
Volume Serial Number is 085D-1A58

Directory of C:\Virtual Machines\Backup\Ubuntu 8.04 JeOS

10/17/2011  02:32 PM  <DIR>          -
10/17/2011  02:32 PM  <DIR>          -
05/05/2008  02:22 PM             1,455  readme.txt
05/05/2008  02:12 PM        186,777,600  root.vmdk
05/05/2008  01:53 PM        131,072     swap.vmdk
05/05/2008  01:52 PM         8,684  Ubuntu.nvram
05/05/2008  01:17 PM              0  Ubuntu.vmsd
05/05/2008  02:13 PM         1,311  Ubuntu.vmx
05/05/2008  01:17 PM          261  Ubuntu.vmxfs
              7 File(s)        186,920,383 bytes
              2 Dir(s)        14,478,630,912 bytes free

C:\Virtual Machines\Backup\Ubuntu 8.04 JeOS>

```

شکل ۸-۶: لیست فایل ها در ماشین مجازی VMware

**Vmdk:** هارد مجازی است و حداکثر اندازه‌ی آن ۲ GB است. به علاوه فایل‌های vmdk حاوی داده‌های ماشین مجازی به اضافه‌ی فضای مورد نیاز سربر است. فایل‌های Vmdk در نسخه‌های پیشین این نرم‌افزار به نام فایل dsk بودند.

**Xnra: Bios:** ماشین مجازی و تعداد هارد درایوهاست.

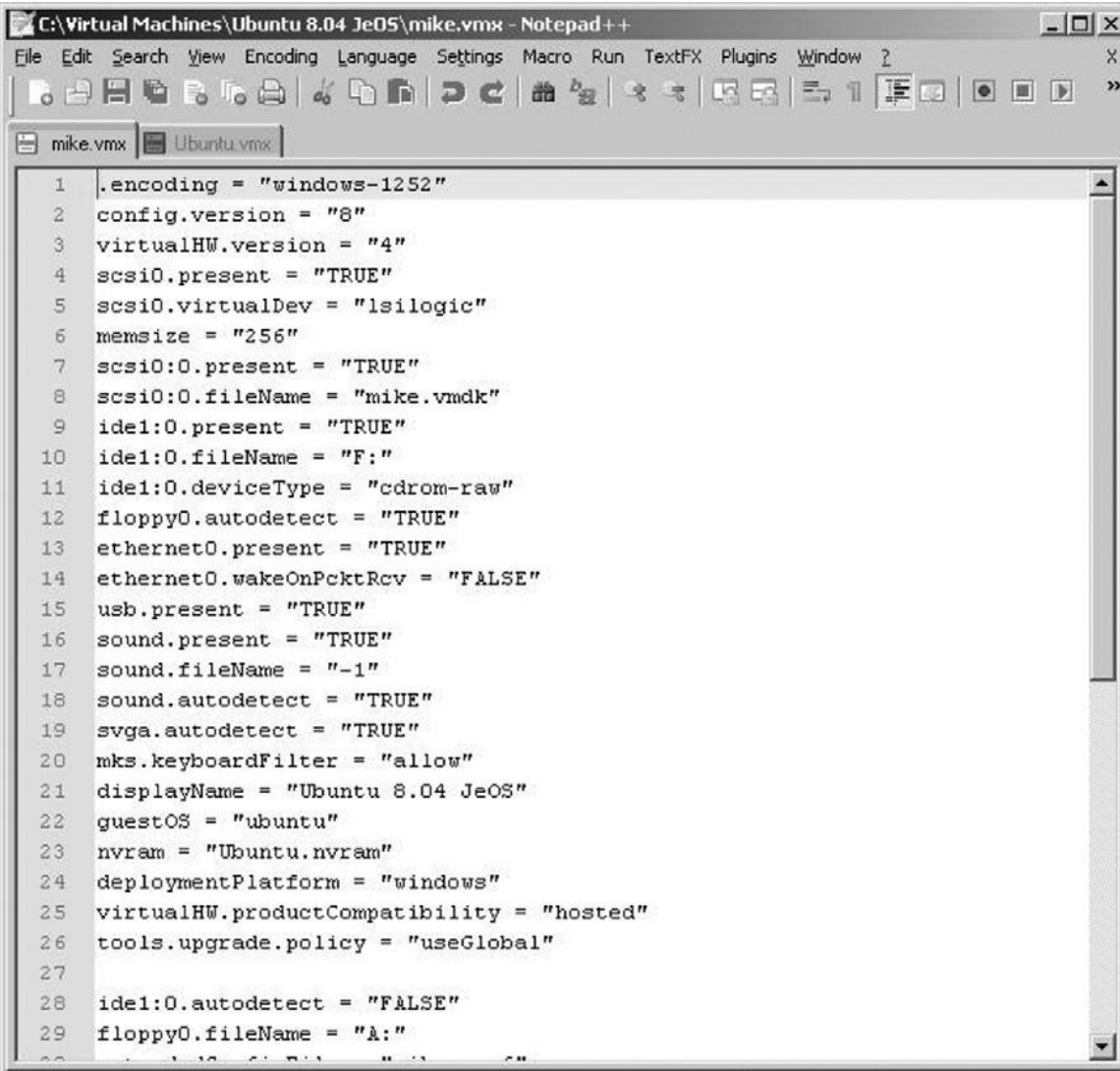
**vmsd:** فایل محل ذخیره‌ی حالت جاری ماشین مجازی و ابرداده‌های snapshot.

**Vmx:** این فایل متنی پیکربندی ماشین مجازی را در خود ذخیره می‌کند و شامل اطلاعاتی در مورد سیستم‌عامل، ابزارهای متصل به آن، ارتباطات شبکه و... است. در شکل ۸-۷ محتویات فایل پیکربندی نمونه را مشاهده می‌کنید.

**Vmxfs:** فایل دیگری شامل ابرداده‌هایی در خصوص سیستم‌عامل ماشین مجازی است که به وسیله‌ی گروهی از کاربران به کار می‌رود.

## پنهان سازی داده ها در فایل های VMware Image

اکنون که با فایل‌هایی که ماشین مجازی VMware Image را شکل می‌دهند آشنا شدیم، می‌توانیم بر روی فایل حاملی که مناسب به پنهان سازی حجم بزرگی از داده‌ها هستند تمرکز کنیم. از آن جایی که فایل هارد درایو مجازی vmak تنها فایل با اندازه‌ی بزرگ است، به بررسی آن می‌پردازیم. اندازه‌ی این فایل از 50 MB تا 2 GB متغیر است و در محیط‌های بزرگ می‌تواند به اندازه‌های ترابایتی هم برسد، ولی واقعیت آن است که برای مثال ما، فایل به اندازه‌ای که بتوان بر روی شبکه قابل انتقال بوده و بتوان آن را دانلود کرد، مناسب است، ولی در عمل، فایل با اندازه 50 MB بسیار ایده آل است.



```

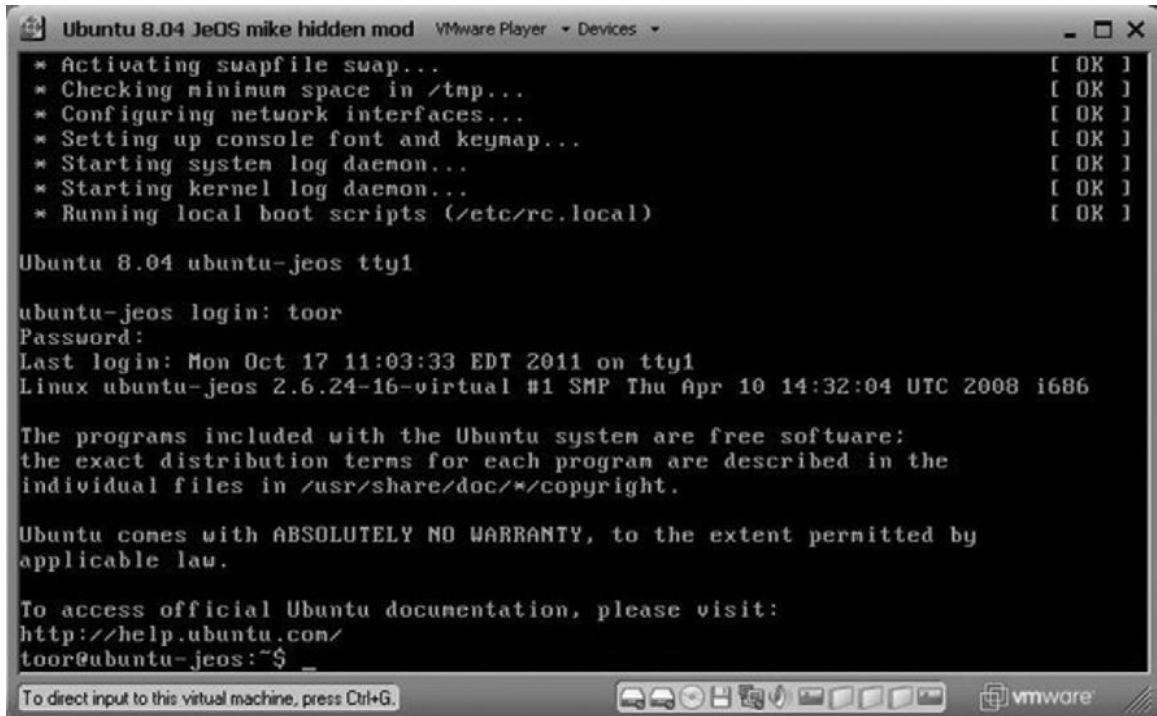
1 .encoding = "windows-1252"
2 config.version = "8"
3 virtualHW.version = "4"
4 scsi0.present = "TRUE"
5 scsi0.virtualDev = "lsilogic"
6 memsize = "256"
7 scsi0:0.present = "TRUE"
8 scsi0:0.fileName = "mike.vmdk"
9 ide1:0.present = "TRUE"
10 ide1:0.fileName = "F:"
11 ide1:0.deviceType = "cdrom-raw"
12 floppy0.autodetect = "TRUE"
13 ethernet0.present = "TRUE"
14 ethernet0.wakeOnPcktRcv = "FALSE"
15 usb.present = "TRUE"
16 sound.present = "TRUE"
17 sound.fileName = "-1"
18 sound.autodetect = "TRUE"
19 svga.autodetect = "TRUE"
20 mks.keyboardFilter = "allow"
21 displayName = "Ubuntu 8.04 JeOS"
22 guestOS = "ubuntu"
23 nvram = "Ubuntu.nvram"
24 deploymentPlatform = "windows"
25 virtualHW.productCompatibility = "hosted"
26 tools.upgrade.policy = "useGlobal"
27
28 ide1:0.autodetect = "FALSE"
29 floppy0.fileName = "A:"

```

1281 chars 1379 bytes 56 Ln: 1 Col: 1 Sel: 0 (0 bytes) in 0 ranges Dos\Windows ANSI INS

شکل ۸-۷: محتویات فایل پیکربندی vmx در VMware

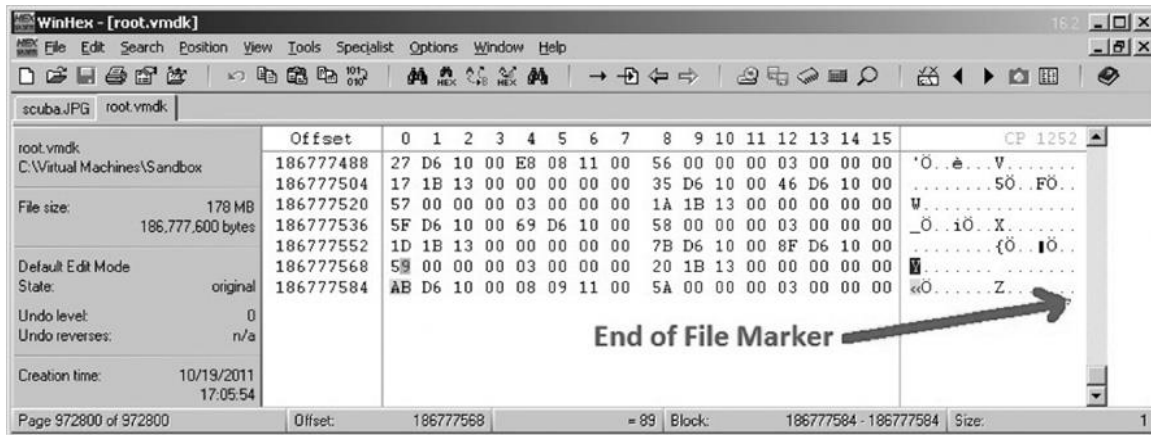
اگر فایل Xwnd را در اختیار ندارید، صدها نرم‌افزار رایگان وجود دارد که برای ساختن این فایل می‌توانید از سایت VMware دانلود کنید. با دانلود مجموعه فایل‌های مورد نیاز که شامل فایل‌های لازم برای اجرای ماشین مجازی به همراه فایل دیسک مجازی Xvmdk هم می‌شود کار خود را با VMware player شروع می‌کنیم (عکس ۸-۸).



شکل ۸-۸: اجرای سیستم عامل ابنتو ۸ به شکل ماشین مجازی

پیش از پنهان‌سازی داده‌ها در فایل vmdk، باید مطمئن شویم که ماشین مجازی خاموش بوده و در حال اجرا نباشد؛ در غیر این صورت ممکن است تأثیر وارونه داشته باشد. اگر ماشین مجازی خاموش است، می‌توان فایل xwnd را به وسیله‌ی v/:Hen برای تحلیل و مشخص کردن مکان مناسب پنهان-سازی داده‌ها باز کنید.

مقادیر Header در فایل vmdk گوناگون است، اما رایج‌ترین آن‌ها KDMv (از راست به چپ خوانده می‌شود) است که در فایل‌های mf پیدا می‌شود. mf فایل‌های دیسک‌های مجازی هستند که لزوماً همگی به صورت یک فایل واحد هستند و همچنین محتویات فایل vmdk فایلی است که جزئیات Vdka دیسک، هندسه دیسک (شبیه هندسه فیزیکی دیسک اصلی) اندازه‌ی image مجازی دیسک و محل قرار گرفتن آن در دیسک (به وسیله‌ی تعیین کردن offset) را شامل می‌شود. نکته مهم، حفظ اندازه‌ی فایل vmdk برای پیشگیری از خطای ناسازگاری در زمان اجرای ماشین مجازی است. از آنجایی که فایل vmdk نسخه‌ی مجازی از هارد دیسک فیزیکی است، پس مثل هارد دیسک فیزیکی شامل pad سکتور و پارتیشن می‌باشد؛ در نتیجه بسیاری از نشانگرهای pad را در دیسک مجازی، به ویژه در پایان آن می‌توان مشاهده نمود (عکس ۸-۹).



شکل ۸-۹: pad موجود در دیسک مجازی که با صفر نشان داده شده است

در این مثال، پنهان‌سازی یک فایل jpeg به اندازه‌ی دو MB استفاده کرده و آن را در دیسک مجازی vmdk درج می‌کنیم. برای اضافه کردن فایل عکس به این فایل، در واقع ۲MB از داده‌ای pad را با ۲MB محتویات فایل عکس جایگزین می‌کنیم و این کار را با استفاده از winHex برای کپی فایل عکس مورد نظر در فایل xvmdk و جایگزینی داده‌های مربوطه به وسیله‌ی عملکرد replace انجام می‌دهیم. اما ابزار ویرایش ویژه‌ی فایل vmdk هم وجود دارد.

Dsfok-tools مجموعه‌ای از نرم‌افزارهای تحت ویندوز است که بدون باز کردن فایل در ویرایشگر ویندوز، امکان ویرایش فایل vmdk را فراهم می‌کند. معمولاً از این ابزارها برای ویرایش نشانه‌گر فایل vmdk استفاده می‌شود و شامل برنامه‌های dsfigDsfo می‌باشد. برنامه dsdo اجازه‌ی مشاهده‌ی اطلاعات فایل vsk را داده و از دستور dsf می‌توان برای تزریق اطلاعات به فایل vsk استفاده کرد. از آنجایی که می‌خواهیم فایل jpeg خود را به فایل حامل VMDK اضافه کنیم از دستور dsfi برای تغییر داده‌های موجود در فایل به جای اضافه کردن فایل jpeg به خود فایل و افزایش طول آن که می‌تواند باعث ناسازگاری و خراب شدنش شود استفاده می‌کنیم. به علاوه تغییر اندازه‌ی vmdk می‌تواند باعث خطا در زمان اجرا گردد و این به خاطر ناسازگاری اندازه‌ی فایل با پارامتر متناظر آن در توصیف‌گر فایل می‌باشد.

فرم دستوری dsfi به شکل زیر است:

<dsfi <destination> <offset> <size> <source>

اندازه‌ی null به بیشینه‌ی اندازه‌ی ممکن تعبیر می‌شود.

اندازه منفی بر اساس اندازه‌ی فعلی فایل محاسبه می‌گردد.

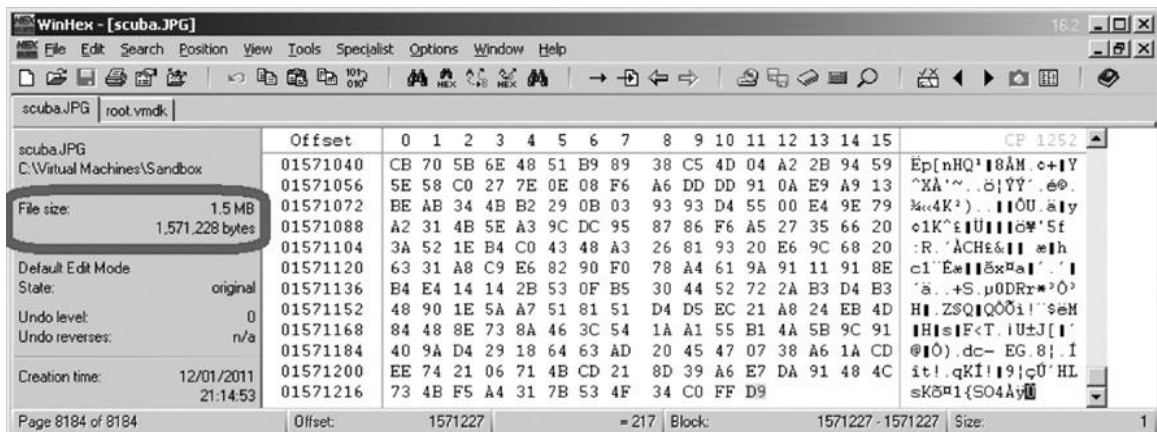
Offset منفی از انتهای فایل محاسبه می‌شود.



استفاده از «e» به عنوان offset، پایان فایل را نشان می دهد.

استفاده از «\$» به عنوان مقصد، فقط باعث بررسی امضا MD5 می شود.

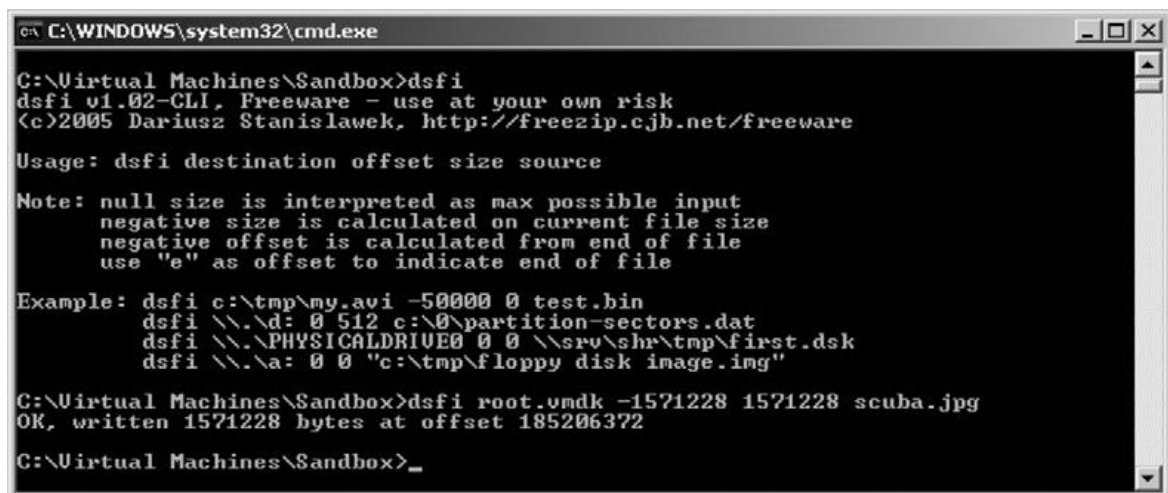
برای تزریق فایل jpeg، از نشانگر پایان فایل (Eof) شروع کرده و به اندازه ی مورد نیاز به عقب برمی گردیم. بنابراین باید نخست اندازه ی فایل jpeg را مشخص کنیم. پس آن را در winHex باز کرده و اندازه ی خام فایل را در سمت راست و به مقدار ۱۰۵۷۱ بایت مشاهده می کنیم (شکل ۸-۱۰).



شکل ۸-۱۰: payload: اندازه فایل jpeg

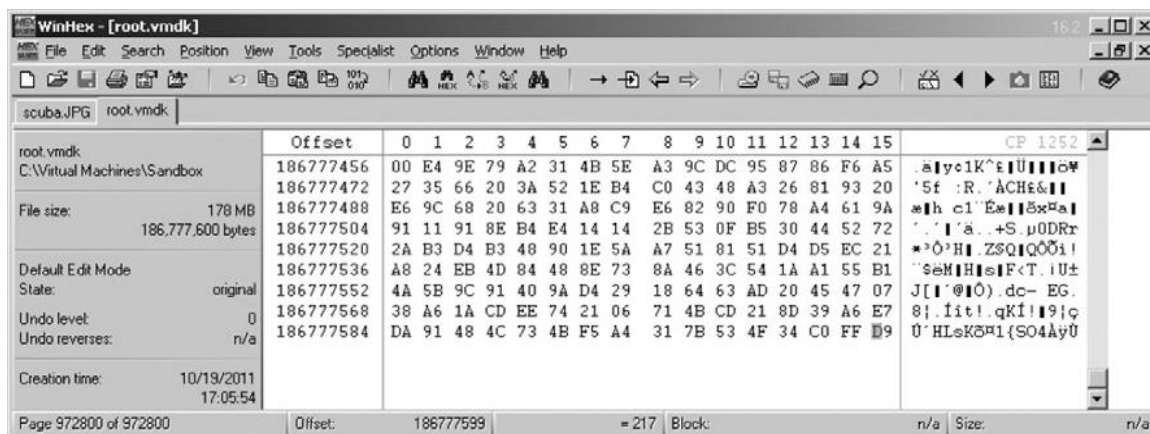
اکنون با استفاده از دستور Dsfi می توانیم فایل jpeg را داخل فایل vmdk اضافه کنیم. از offset منفی برای محاسبه ی آدرس، با شروع از آخر فایل استفاده می کنیم. اگر بخواهیم فایل را تحت نام root.vmdk ذخیره کنیم (عکس ۸-۱۱)، دستور به شکل زیر است:

C:\ dsfi root.vmdk -1571228 1571228 scuba.jpg



شکل ۸-۱۱: استفاده از ابزار dsfi برای افزودن payload به فایل حامل VMDK

اگر فایل `vout.vmdk` را دوباره باز کنیم، مشاهده می‌کنیم که فایل `jpeg` به وسیله‌ی جایگزین کردن ۱۰۵۵ بایت از محتویات `root.vmdk` با محتویات فایل `jpeg` در آن درج شده است (عکس ۸-۱۲).



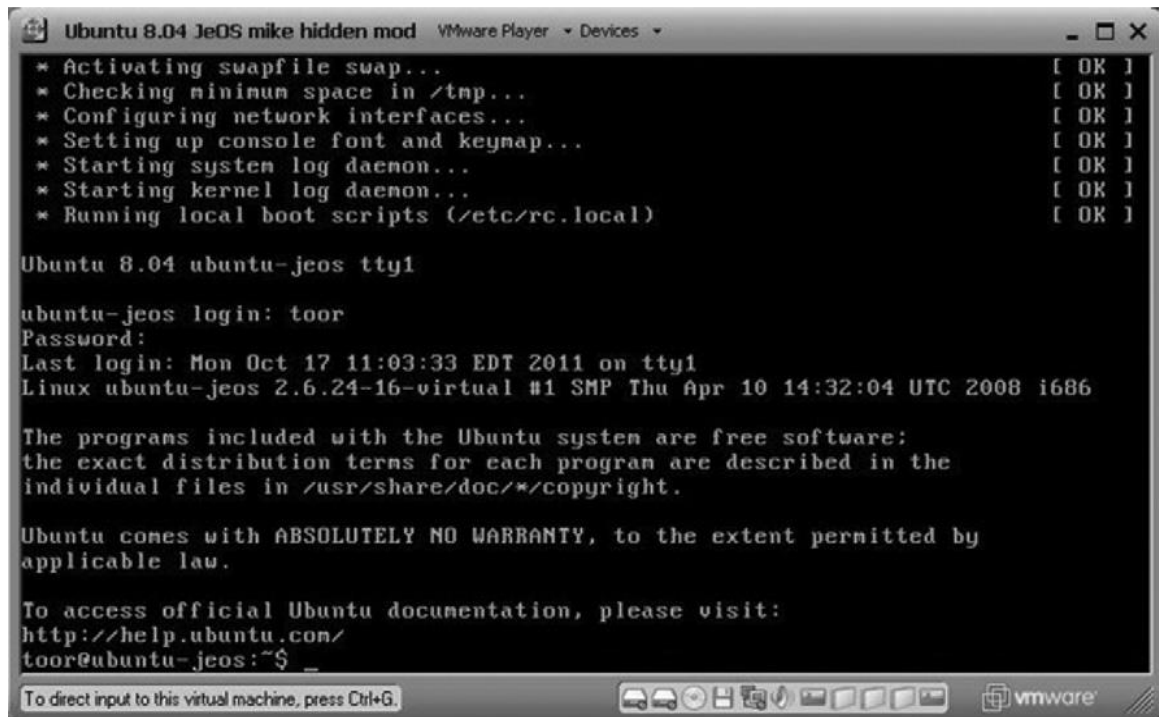
شکل ۸-۱۲: فایل `jpeg` که به فایل `root.vmdk` درج شد.

اکنون فایل دیسک مجازی داریم که درون آن داده‌هایی را جاسازی کرده‌ایم و می‌توانیم به عنوان پیکربندی آماده استفاده `dead drup` در شبکه `upload` تا بعداً توسط گیرنده برداشته شود.

نکته d جالب در مورد فایل `VMDK` این است که به رغم این تغییرات، فایل همچنان توسط `VMV player` قابل اجرا بوده و کماکان عملکرد نرمالی همانند یک ماشین مجازی دارد و هیچ نشانی از تغییر از خود بروز نمی‌دهد. هیچ خطایی به کاربر داده نمی‌شود و حتی پیامی هم به وی نشان داده می‌شود که آیا مایل به کپی کردن یا تغییر محل فایل عکس پیش از اجرای ماشین مجازی است یا خیر؟ و این پیامی است که در مورد عکس‌های دانلود شده معمولاً نشان داده می‌شود. هیچ پیامی هم دال بر تغییر ماشین مجازی از آخرین بار استفاده از آن نشان داده نمی‌شود. تنها اگر فایل `image` پیش و پس از تغییر وجود داشته باشد، می‌توان با مقایسه آن‌ها به وجود فایل `jpeg` اضافه شده پی برد.

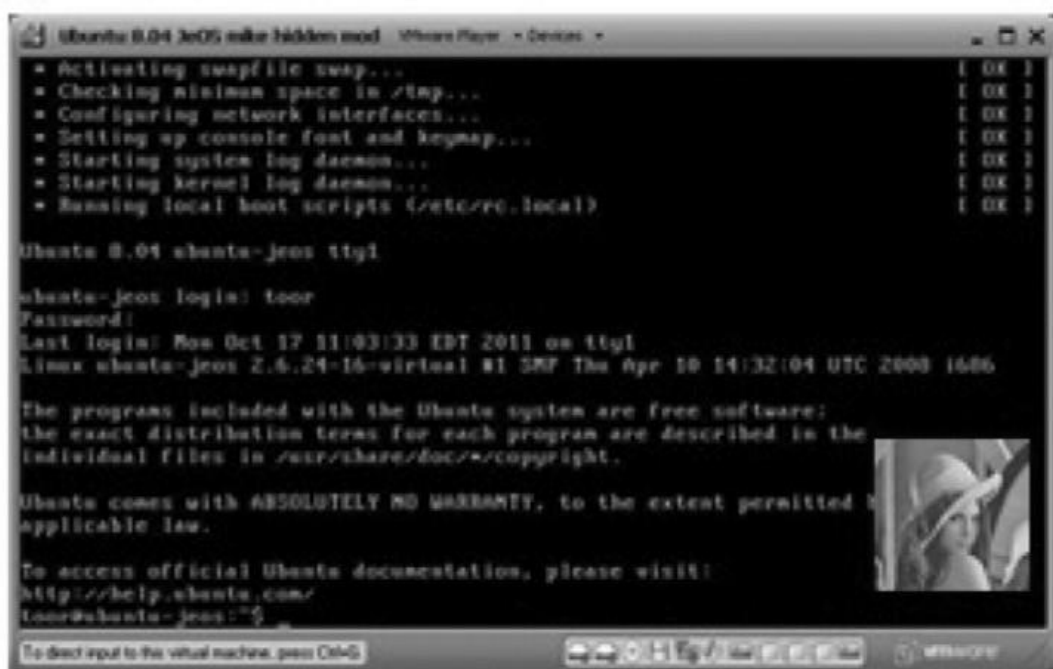
اجراهای پی در پی ماشین مجازی هیچ تأثیری بر عکس پنهان شده ندارد، اما از آنجایی که ما در مورد دیسک مجازی بحث می‌کنیم، در استفاده‌های مکرر از ماشین مجازی، امکان رونویسی داده‌ها وجود دارد. ولی واقعیت این است که در مثال ما کاربر مورد نظر می‌داند که از سازوکار دیسک مجازی به عنوان پوشش استفاده شده است، پس نگرانی در مورد استفاده سایر کاربران و رونویس اطلاعات توسط آن‌ها

وجود ندارد. در مثال زیر ماشین مجازی حاوی داده‌های اضافه شده، هیچ تأثیری بر عملکرد ماشین مجازی از خود نشان نمی‌دهد (عکس ۸-۱۳).



شکل ۸-۱۳: ماشین مجازی VMV پس از افزودن داده‌های پنهان، بدون هیچ مشکلی اجرا می‌شود.

استخراج عکس پنهان شده از فایل vmdk ماشین مجازی vmvov بسیار ساده‌تر از پنهان کردن آن است (عکس ۸-۱۴).



شکل ۸-۱۴: بیرون کشیدن عکس پنهان در فایل ماشین مجازی

برای این کار از ابزار dsfk که پیش‌تر به آن اشاره شد، به شکل زیر استفاده می‌کنیم.

C:\dsfo root.vmdk -1571228 1571228 scuba.jpg

این کار باعث می‌شود آخرین \57/ بایت از فایل استخراج و تحت عنوان su.jpg ذخیره شود.

## چکیده

در این فصل راه‌های پنهان شدن محیط‌های مجازی و روش‌های پنهان‌سازی داده‌ها در ماشین مجازی را شرح دادیم. اگرچه این راه‌های پنهان‌سازی ممکن است با استفاده از روش‌های دیگری باشد که در این کتاب آن‌ها را بررسی کردیم، اما بسیار مستعد و ناپیدا<sup>۱</sup> در ذات خود هستند. استفاده از محیط‌های مجازی برای پنهان‌سازی داده در پیاده‌سازی راهبرد امنیت شبکه‌ها باید مورد توجه ویژه قرار گیرد، تا قابل تشخیص بوده و از استفاده نامطلوب از شبکه پرهیز شود. برای توسعه دهندگان و کاربران ماشین‌های مجازی در سطح گسترده، یکپارچگی داده‌ها در این سیستم‌ها باید نگرانی جدی تلقی شود.

---

<sup>۱</sup> stealthy

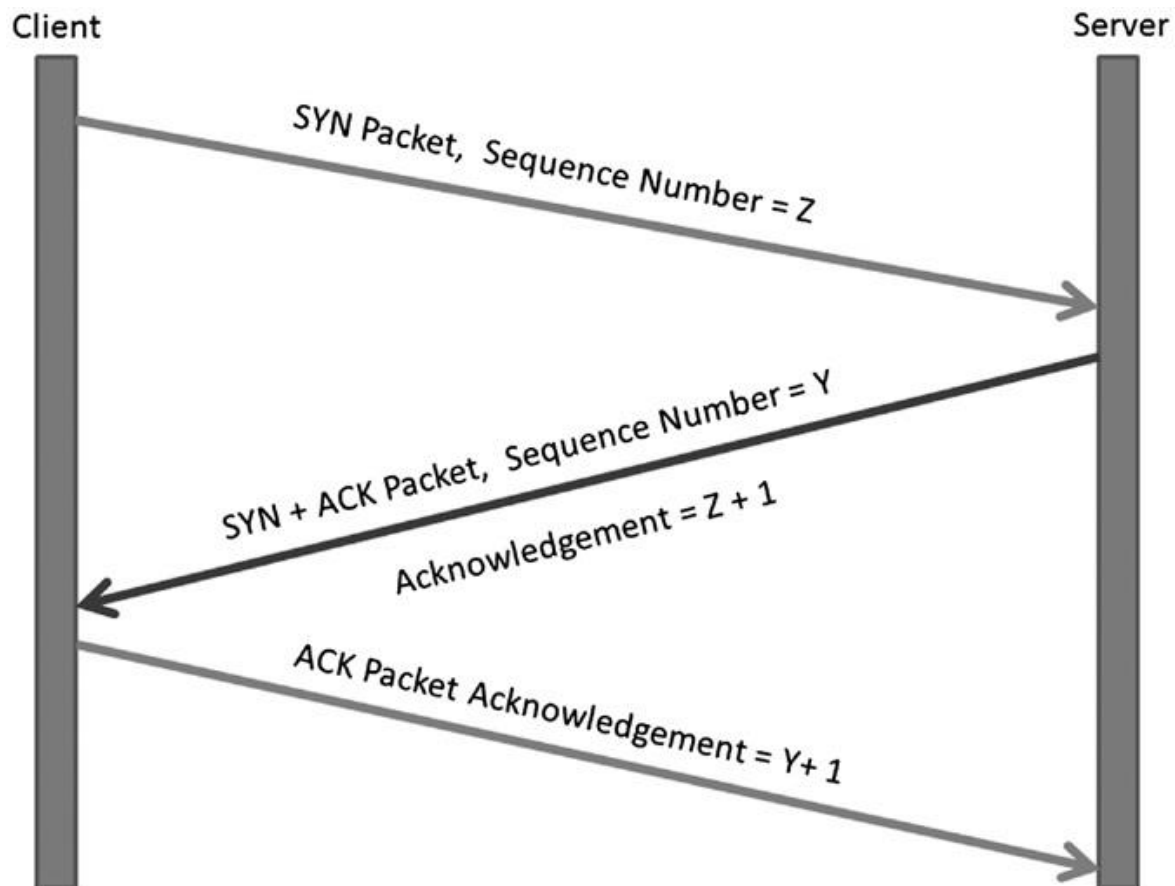
~~~~~

### پنهان سازی داده‌ها در پروتکل‌های شبکه

پس از حمله Cinco de Mayo که در اولین دوشنبه ماه می سال ۱۹۹۷ میلادی به وقوع پیوست، هکرها جزئیات حمله first mondy را به همراه مقاله‌ای تحت عنوان < کانال‌های پویش در مجموعه‌ی پروتکل‌های TCP/IP > نوشته‌ی رولند منتشر کردند، با این مضمون که:

مجموعه پروتکل‌های Tcp/ip نقاط ضعفی دارد که به هکر اجازه می‌دهد تا با استفاده از این نقاط ضعف و استفاده از این پروتکل در قالب کانال پوششی، داده‌ها را به وسیله‌ی packet در شبکه منتقل کنند. این مقاله سعی در ترسیم این نقاط ضعف به شکل مثال‌های عملی و تئوری داشت. روش‌های بیان شده در این مقاله، نقاط ضعفی از پروتکل TCP را آشکار کرد که راه ساده و کارآمدی را در پنهان‌سازی و داده‌ها در مرحله‌ی آغازین برقراری ارتباط TCP را در اختیار مهاجم قرار می‌دهد. مثل بیشتر هشدارهای امنیتی از این نوع در فضای سایبری، صدای اعتراض اولیه برای طراحی پروتکل‌هایی که مراحل آغازین امن‌تری دارد و می‌تواند چنین حملاتی را خنثی کند، به گوش می‌رسد. پس از مدتی و با کاهش اعتراض‌ها توجه خود را به حملات بعدی معطوف کرده و بسیاری از ما آسیب‌پذیری از این ناحیه را به فراموشی می‌سپارند.

برای نشان دادن انعطاف‌پذیری واقعی در استفاده از پروتکل TCP به عنوان پوشش، Craig را تغییر داده و برای پنهان‌سازی داده‌ها، مقدار شماره توالی که خودمان مقداردهی کرده‌ایم را به کار بردیم. اجازه دهید نگاه دقیق‌تری به چگونگی عملکرد TCP در قالب پوششی داشته باشیم. شکل ۹-۱ مراحل سه‌گانه‌ی hand shaking لازم برای ایجاد ارتباط TCP را نشان می‌دهد. زمان برپایی ارتباط از نوع TCP عنصر حساس، انتخاب شماره‌ی بسته‌های متوالی در آغاز ارتباط است.



شکل ۹-۱: مراحل سه‌گانه Hand shake در برپایی انتقال TCP

از آنجایی که آغازگر ارتباط TCP (در مثال ما client) مقدار اولیه‌ی شماره سریال بسته‌ها را مشخص می‌کند، در نتیجه، چیزی را در اختیار داریم، که بدان نیاز داریم یعنی راهی برای انتقال اطلاعات به وسیله‌ی شماره سریال بسته‌های ارسالی. در ارتباط TCP مراحل شروع برقراری ارتباط را به وسیله‌ی نرم‌افزار wire shark در شکل ۹-۲ نشان داده‌ایم. شماره سریال مشخص شده به وسیله‌ی client را در مبنای ۱۶ و معادل آن را در مبنای ده در پایین صفحه مشخص کرده‌ایم. در واقع می‌بایست مقدار مبنای ۱۶ را معادل مبنای ده بر اساس قاعده small ending به شکل زیر تبدیل کنیم که معادل ۷۴۲۰۸۴ در مبنای ده است. برای مبهم کردن اطلاعات پنهان شده، در این مثال عدد ثابت  $k$  را برابر ۶۲۳۶ که بیشتر بین گیرنده و فرستنده به اشتراک گذاشته شده را در عدد بالا ضرب می‌کنیم. بنابراین هر بایت اطلاعات را که خواستیم ارسال کنیم، کد اسکی مربوطه را در ۶۲۳۶ ضرب می‌کنیم.

$$\text{ASCII value} \times K = \text{Sequence Number}$$

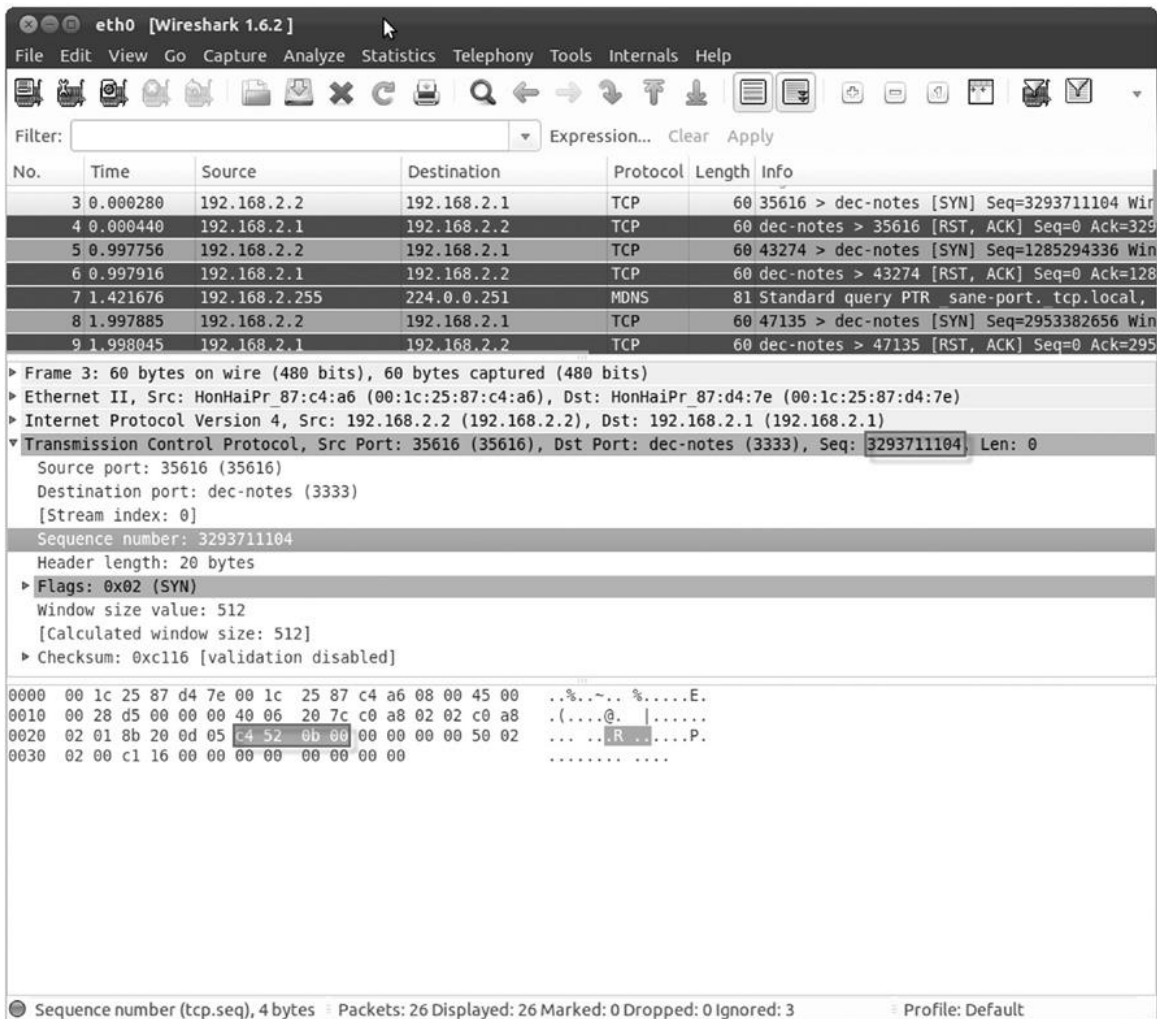
برای آشکارسازی نویسه‌ی پنهان در سمت گیرنده، پروسه وارون را انجام دهید.



## Sequence Number $K = \text{ASCII value}$

برای مثال در شکل ۹-۲ می‌خواهیم حرف W با کد اسکی ۱۱۹ را ارسال نماییم.

Decimal = 00 0B 52 C4 in hex  $742,084 = 6236 \times 119$



شکل ۹-۲: مشاهده‌ی مراحل Hand shake در برقراری ارتباط TCP به وسیله Wireshark

بنابراین با استفاده از فرمول بالا چگونگی عملکرد ارسال کامل پیام در شکل ۹-۳ نشان داده شده

است.

```

snort-mailserver@SNORT-MailServer: ~/Desktop/Covert_TCP
File Edit View Search Terminal Help
Covert TCP 1.0 (c)1996 Craig H. Rowland (crowland@psionic.com)
Not for commercial use without permission.
Destination Host: 192.168.2.1
Source Host      : 192.168.2.2
Originating Port: random
Destination Port: 3333
Encoded Filename: data.txt
Encoding Type    : IP Sequence Number
Client Mode: Sending data.

Sending Data: 0 (encoded), w (unencoded)
Sending Data: L (encoded), e (unencoded)
Sending Data: 0 (encoded), t (unencoded)
Sending Data: T (encoded), s (unencoded)
Sending Data: 0 (encoded), t (unencoded)
Sending Data: 0 (encoded), o (unencoded)
Sending Data: 0 (encoded), n (unencoded)
Sending Data: L (encoded), e (unencoded)
Sending Data: 0 (encoded),
(unencoded)
snort-mailserver@SNORT-MailServer:~/Desktop/Covert_TCP$

```

شکل ۹-۳: پنهان‌سازی داده‌ها در شماره سریال بسته‌های TCP

می‌توانیم روش بالا را در قالب پوشش گسترش داده و مثلاً از XOR مقادیر برای پوشش ارسال کاراکتری که قصد ارسال آن را داریم، استفاده کنیم. با این کار ویژگی تصادفی بودن شماره توالی هم بهبود می‌یابد. این روش پنهان‌سازی داده‌ها محدودیت ارسال یک یا دو بایت را در هر بسته دارد، لیکن اگر شما بخواهید کلیدی را مبادله کنید یا پیام کوتاهی بفرستید، یا فقط برای اعلام حضور deacom استفاده کنید، این روش پنهان‌سازی در میان میلیاردها بسته‌ای که هر روزه حتی در سازمان‌های کوچک تولید می‌شود، به راحتی قابل اجراست.

جای تعجب است اگر شرکت سیسکو راه‌حلی برای گریز از حمله دوشنبه ۵ می ۲۰۱۴ در صورت تکرار دوباره‌ی آن داشته باشد. حتی امروزه نه تنها در پروتکل TCP بلکه در صدها پروتکلی که هر روزه به کار می‌روند نقاط ضعفی وجود دارد که بیشتر آن‌ها را مستعد پنهان‌سازی داده‌ها بدون آسیب به عملکردشان می‌کند.

## پنهان‌سازی داده‌ها در VOIP

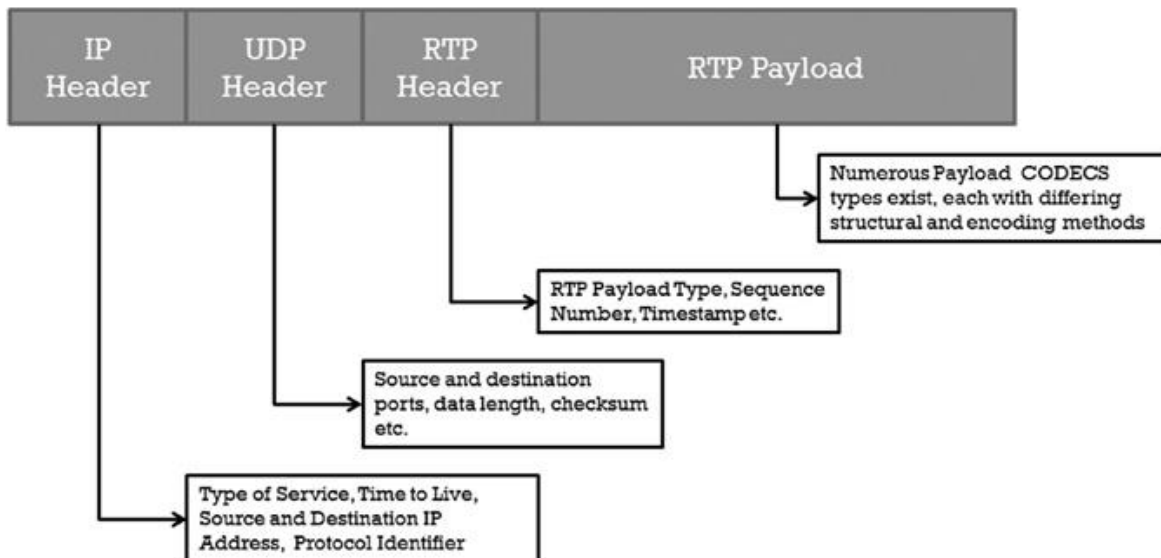
یکی از حوزه‌های جالب پنهان‌سازی داده‌ها، فناوری VOIP است. علت هم بسیار آشکار است. VOIP به شکل گسترده‌ای کاربردی بوده و از آن استفاده می‌شود. VOIP تعداد زیادی بسته کوچک تولید می‌کند

که برای پنهان کردن بخش‌های کوچک از پیام طولانی، بسیار ایده‌آل و مناسبی است. با توجه به گوناگونی گسترده‌ی نوع بسته‌ها، انواع کدها و روش‌های کدگذاری در VOIP، آن را به پوشش مناسبی بدل کرده که کشف آن مثل پیدا کردن سوزن در انبار کاه است.

VOIP سازوکارهایی از ارسال بسته‌ها را در سطح شبکه به کار می‌گیرد که اتکا ناپذیرند. مثلاً پروتکل RTP و UDP بسته‌های گم شده یا بسته‌هایی که دیر به مقصد می‌رسند را دوباره ارسال نمی‌کنند. در نگاه اول ممکن است این روش برای مقاصد پنهان‌سازی داده مناسب به نظر نرسد، زیرا گم شدن بسته‌ای که بخشی از پیام سری را در خود پنهان کرده، به ویژه اگر رمزنگاری هم شده باشد، مشکل بزرگی در دریافت کل پیام ایجاد می‌کند. ولی یکی از روش‌هایی که ما بررسی می‌کنیم، استفاده از این نقطه ضعف و تبدیل آن به نقطه‌ی قوت است.

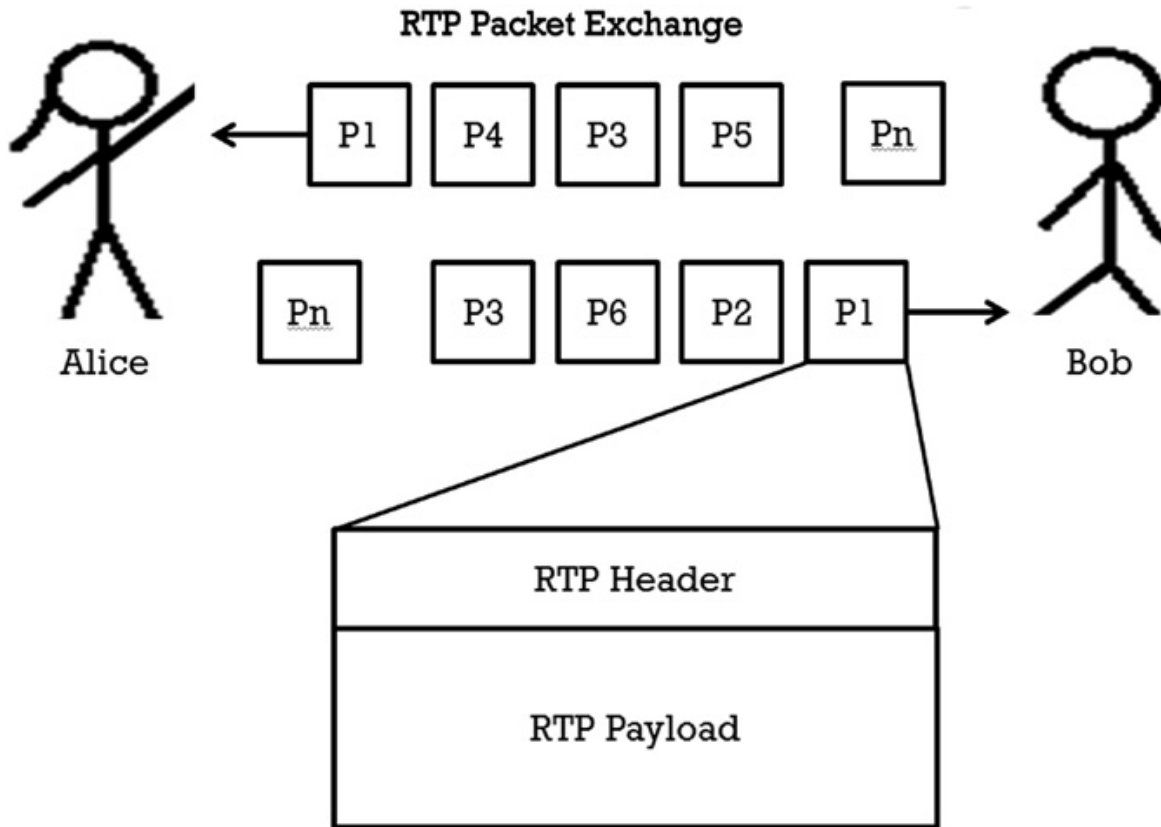
برای تشریح عناصر اصلی در روش‌های پنهان‌سازی داده‌ها در روش RTCP، کار خود را با بررسی پروتکل ساده‌ی RTC که از پروتکل UPC در برقراری ارتباط نقطه به نقطه استفاده می‌کند آغاز می‌کنیم. آشکار است که برای استفاده از این روش پنهان‌سازی در پیکربندی واقعی VOIP به پروتکل‌های SIP و RTCP و ... هم نیاز دارید. شکل ۴-۹ ساختاری ساده از فرصت بسته RTP VOIP را نشان می‌دهد. همان گونه که مشاهده می‌کنید بسته RTP در بخش payload بسته UPP جای می‌گیرد که خود بسته UPP در داده‌ها را در ارتباط اتکا ناپذیر حمل می‌کند و خود بسته UDP، به نوبه‌ی خود در دنباله‌ی payload بسته‌ی IP وظیفه‌ی مسیریابی مورد نیاز در اینترنت را برای رسیدن به مقصد فراهم می‌کند.

### VoIP RTP Packet Structure



شکل ۴-۹: ساختار ساده شده‌ی بسته VoIP RTP

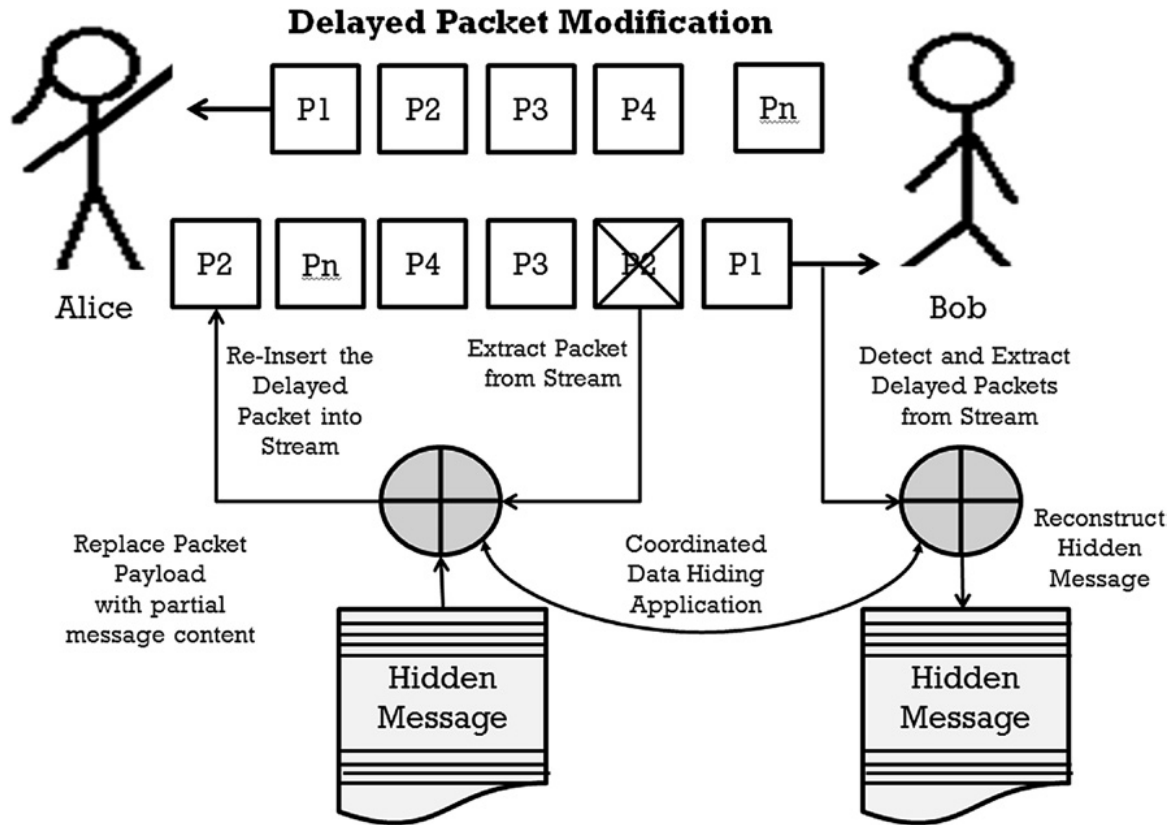
در شکل ۵-۹ به چگونگی تبادل بسته‌های RTP بین دو شخص Alice و Bob توجه نمایید.



شکل ۵-۹: بسته‌های RTP مبادله شده بین آلیس و باب

همان‌گونه که در شکل ۵-۹ مشاهده می‌کنید، A و D توالی بسته‌های RTP را در جریانی پیوسته از داده‌ها با هم مبادله می‌کنند. اگر به یاد داشته باشید، بسته‌های دریافتی ممکن است نادیده گرفته شوند یا برخلاف ترتیب اولیه ارسال، به مقصد برسند، یا با تأخیر به مقصد برسند، زیرا پروتکل UDP ارتباط قابل اتکا و تحویل بسته‌ها در توالی ارسال را گارانتی نمی‌کند. همان‌گونه که پیشتر اشاره شد، بسته‌های کنترلی و سایر فیلدهای کنترلی می‌توانند در برقراری دوباره‌ی ارتباطی مثلاً به طول سه هزار کیلومتر که قطع شده، کمک کنند.

ساده‌ترین روش پنهان‌سازی داده‌ها افزودن مستقیم آن‌ها به بخش داده‌ی Payload بسته‌هاست. پیام سری به بخش‌های کوچکی تقسیم شده و در payload جاسازی می‌شود. این روش یکی از دو روش پایه‌ی پنهان‌سازی داده‌ها است (عکس ۵-۹).



شکل ۹-۶: درج داده‌ها در بخش payload

(۱) بخش کوچکی از پیام سری در بخش header قسمت payload بسته RTP درج می‌شوند. از آنجایی که بیشتر انواع RTP pay، هدر از پیش تعیین شده‌ای دارند که اطلاعاتی را به گیرنده در خصوص پیکربندی payload منتقل می‌کنند، امکان افزودن تعداد کمی بایت در این قسمت وجود دارد.

(۲) بخش کوچکی از پیام به جریان از بیتی‌ها تبدیل شده و هر بیت با کم‌ارزش‌ترین بیت بخش داده‌ها payload جایگزین می‌گردد.

در شکل ۹-۷ یک بسته از بسته‌های مبادله شده به وسیله‌ی پروتکل RTP VOIP بین دو میزبان را مشاهده می‌کنید. این بسته یکی از هزاران بسته‌ی رد و بدل شده بین دو رایانه محلی با آدرس ۱۹۲،۱۶۸،۲،۲ و ۱۹۲،۱۶۸،۲،۱ است و برای مشاهده‌ی داده‌های پنهان شده در این بسته، پیام کوتاه Data Hiding را درج کردیم. این پیام می‌تواند نخست رمزنگاری شده یا به جریان بیتی تبدیل شود و سپس هر بیت از آن را جایگزین مقدار کم‌ارزش‌ترین بیت محتویات بسته VOIP کرده تا دست‌کم به صورت مجازی غیرقابل تشخیص گردد. در میان هزاران بسته رد و بدل شده بین میزبان‌ها و بهترین حالت، کشف این بسته‌ها کار مشکلی است.

Wireshark 1.6.5 [SVN Rev 40429 from /trunk-1.6]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply

| No. | Time     | Source      | Destination | Protocol | Length | Info                                                         |
|-----|----------|-------------|-------------|----------|--------|--------------------------------------------------------------|
| 205 | 2.975544 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59232, Time=24000 |
| 206 | 2.976206 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59232, Time=24000 |
| 207 | 3.005577 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59233, Time=24240 |
| 208 | 3.006237 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59233, Time=24240 |
| 209 | 3.034095 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59234, Time=24480 |
| 210 | 3.034757 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59234, Time=24480 |
| 211 | 3.064103 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59235, Time=24720 |
| 212 | 3.064785 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59235, Time=24720 |
| 213 | 3.094779 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59236, Time=24960 |
| 214 | 3.095654 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59236, Time=24960 |
| 215 | 3.124423 | 192.168.2.2 | 192.168.2.1 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59237, Time=25200 |
| 216 | 3.125702 | 192.168.2.1 | 192.168.2.2 | RTP      | 294    | PT=ITU-T G.711 PCMA, SSRC=0xDDEE0EE8F, Seq=59237, Time=25200 |

Packet 215 (3.125702) Details:

Time to live: 64  
 Protocol: UDP (17)  
 Header checksum: 0xb481 [correct]  
 Source: 192.168.2.2 (192.168.2.2)  
 Destination: 192.168.2.1 (192.168.2.1)  
 User Datagram Protocol, Src Port: 6000 (6000), Dst Port: 6000 (6000)  
 Real-Time Transport Protocol  
 Stream setup by SDP (frame 5)  
 10... = Version: RFC 1889 Version (2)  
 ..0... = Padding: False  
 ...0... = Extension: False  
 ....0000 = Contributing source identifiers count: 0  
 0... = Marker: False  
 Payload type: ITU-T G.711 PCMA (8)  
 Sequence number: 59236  
 [Extended sequence number: 59236]  
 Timestamp: 24960  
 Synchronization Source identifier: 0xddee0ee8f (3739283087)  
 Payload: 34819d7b45000000080045000041e35540004006435f0a00...

0000 00 1c 25 87 d4 7e 00 1c 25 87 c4 a6 08 00 45 00 ..%...%....E.  
 0010 01 18 00 00 10 00 40 11 b4 81 c0 a8 02 02 c0 a8 ...000...  
 0020 02 01 17 70 17 70 01 04 e9 1e 80 08 e7 64 00 00 ...p.p...d...  
 0030 61 80 de e0 ee 8f 34 81 9d 7b 45 00 00 00 08 00 a...4...[E...  
 0040 45 00 00 41 e3 55 40 00 40 06 43 5f 0a 00 00 01 E...A...C...  
 0050 0a 00 00 02 d1 51 0d 05 36 51 61 5f 9c 9d 99 31 ....Q...6Qa...1  
 0060 80 18 00 28 87 58 00 00 01 01 08 0a 07 d2 83 62 ...(.x.....b  
 0070 07 d2 7f 3c 44 61 74 61 20 48 69 64 69 6e 67 2e ...data Hiding...  
 0080 0a c5 5a 7a 6e 6e 66 4a d5 d5 f5 e1 97 99 87 80 ...Zznrfj...  
 0090 8c 8c 87 fd 12 04 1a 1e 1a 05 1e 6a c5 93 9b 9f .....j...  
 00a0 e9 f3 c5 d5 7a 16 1a 07 1a 10 62 7a 66 7a 5a f5 ...z...bzfzZ.  
 00b0 fd c5 5a 72 72 5a d5 c5 c5 f5 e5 93 9f 85 87 ...ZrfZ...  
 00c0 84 93 c5 62 62 62 6e 16 10 10 14 66 c5 e5 fd c5 ...bbbn...f...  
 00d0 d5 f5 f5 c5 72 6e 14 14 6e 6e 6a 14 6a 62 7a 4a ...fn...nmj.jbzj  
 00e0 4a 5a 5a d5 c5 fd fd e5 e5 ed 95 91 9d 9b 9b 93 jZz...  
 00f0 e5 4a 72 4a 72 62 6a 16 14 62 4a c5 d5 5a 5a d5 .JrJrbj...bZ...Zz.  
 0100 c5 c5 5a 7a 66 66 66 66 62 6e 6a 6e 66 7a 72 4a ...Zzffff bnjnfzrj  
 0110 d5 c5 fd e1 ed e9 95 91 9d 99 85 84 9f e1 5a 4a .....ZJ  
 0120 5a 4a 7a 6e 14 6a Zzn.j

شکل ۹-۷: مشاهده payload بسته RTP

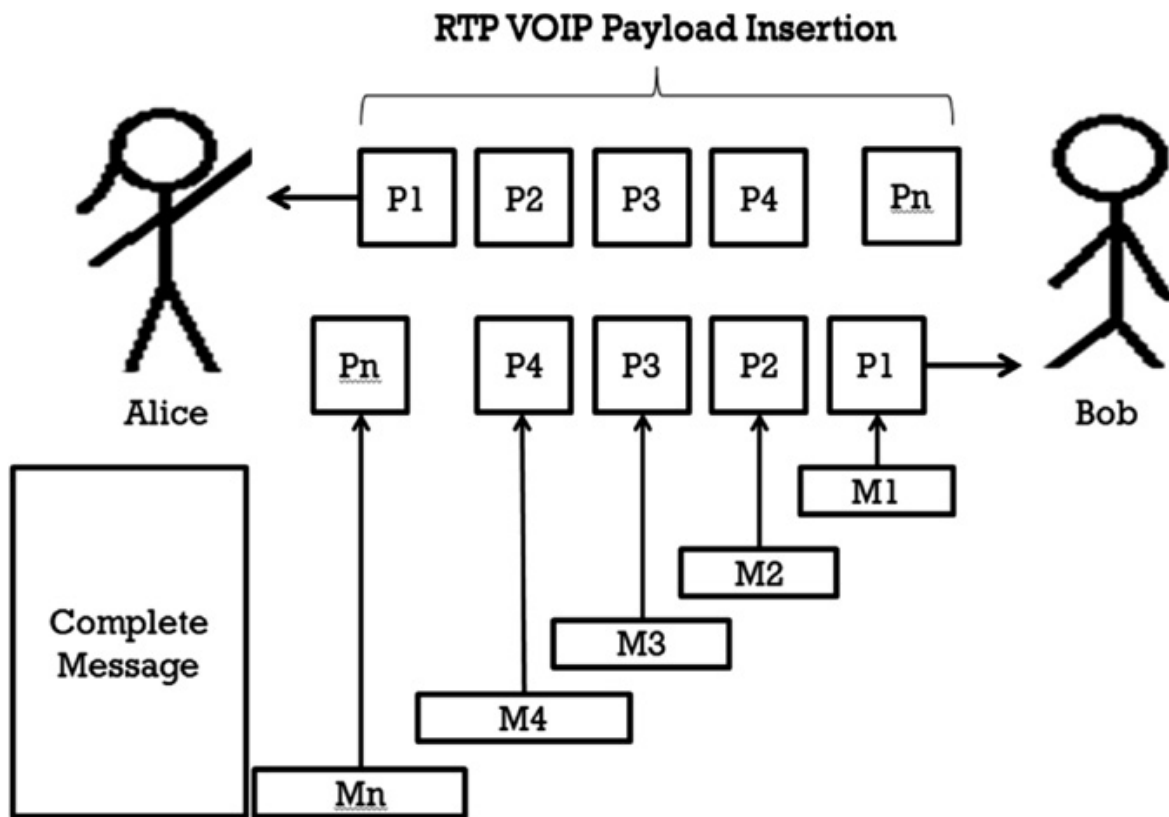
مشکل هر دو روش یادشده این است که اگر گم شود یا دیر برسد، بخشی از پیام پنهان شده هم گم می‌شود. این مشکل را به وسیله‌ی افزودنی در سایر بسته‌ها می‌توان حل کرد. برخی نرم‌افزارها هم امکان افزودگی داده‌ها را می‌دهند و هم می‌توان فرکانس ارسال مجدد بسته‌ها را تعیین کرد.

## شیوه پنهان‌سازی به روش تغییر تأخیر بسته‌ها

این روش در واقع جالب‌ترین روشی است که بررسی می‌کنیم و در شکل ۹-۸ آن را نشان داده‌ایم. این روش از این واقعیت سود می‌جوید، که بسته‌های RTP برخی اوقات دیر می‌رسند یا برابر شماره‌ی توالی در ارسال تحویل گیرنده داده نمی‌شوند یا برخی بسته‌ها در زمان ارسال گم می‌شوند. از این موارد به

عنوان عناصر اصلی در هسته‌ی این روش استفاده می‌شود. این رویکرد به صورت سیستماتیک، برخی بسته‌های را از جریان داده‌ها، پیش از ارسال به وسیله‌ی فرستنده بیرون کنید، که این کار باعث می‌شود تا عمده‌ی برخی بسته‌ها در زمان مقرر تحویل گیرنده نشود و برنامه‌ی گیرنده چه VOIP باشد چه پخش کننده صوتی، جای بسته خالی را بر اساس روش‌های ویژه‌ای پر می‌کنند. از آنجایی که بیشتر بسته‌های RTP محتوا چند هزارم ثانیه از صوت می‌باشد، پس در زمان شنیدن مکالمه یا حتی پخش دوباره‌ی آن، فقدان این بسته‌ها به سختی قابل تشخیص است.

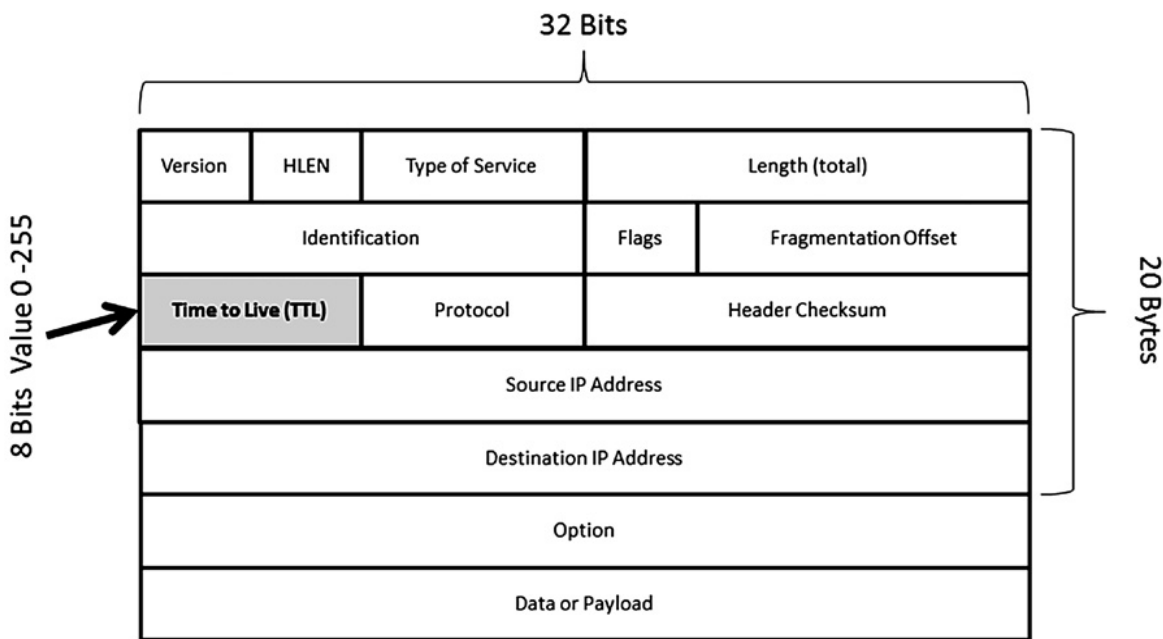
بهترین قسمت داستان اینجاست که وقتی نرم‌افزار پنهان‌سازی داده می‌تواند از تمامی بخش Payload بسته‌هایی که مانع ارسال به موقع آن‌ها شده برای پنهان کردن محتوا پیام متنی استفاده کند. پس از جاسازی داده‌های پنهان، این بسته‌ها به جریان ارسال برگردانده می‌شوند (اما با تأخیر) و برنامه‌گیرنده، از بسته‌هایی که دیر می‌رسند چشم‌پوشی می‌کند (هرگز از آن استفاده نمی‌شود) زیرا جای خالی این بسته با بسته‌های قبلی پر شده است. اما برنامه‌ی پنهان‌سازی در مقصد بسته، تأخیری را تشخیص داده و پس از دریافت آن‌ها، داده‌های پنهان را آشکار و پیام پنهان را بازسازی می‌کند.



شکل ۸-۹: تغییر محتوای برخی از بسته‌های RTP و ارسال آن‌ها با تأخیر

## پنهان‌سازی داده‌ها در لایه IP و در فیلد TTL

از آنجایی که تقریباً هر پروتکلی در هر لایه‌ای مستعد پنهان‌سازی داده‌هاست، تصمیم داریم موتور محرک مسیریابی بسته‌های IP در اینترنت را از دیدگاه پنهان‌سازی داده‌ها بررسی کنیم. پروتکل IP نسخه ۴ را در شکل ۹-۹ مشاهده می‌کنید و هنوز هم هسته‌ی اصلی تحویل بسته‌های TCP و UDP را تشکیل می‌دهد.



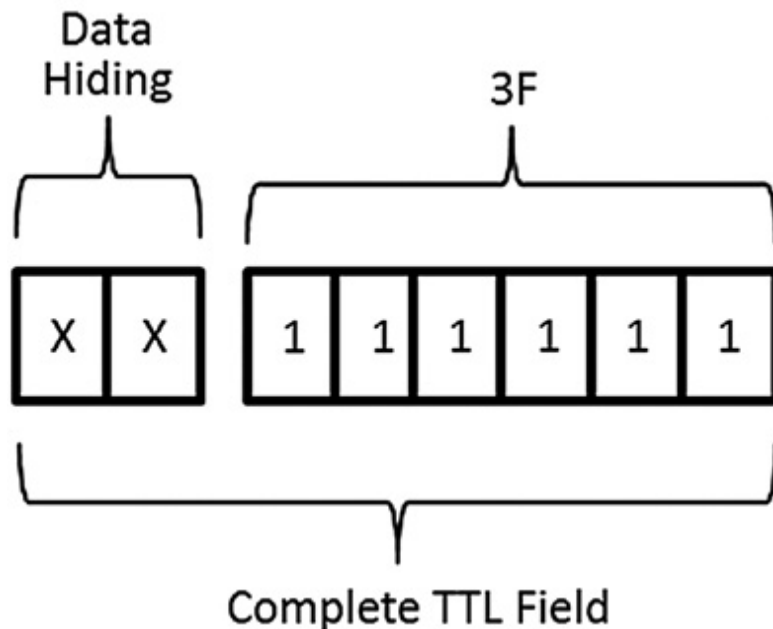
IP Version 4 Format

شکل ۹-۹: دیاگرام بسته IPV4

با بررسی فیلدهای تشکیل دهنده‌ی هدر آن پی می‌بریم که هر بسته‌ی استاندارد ۲۰ بایت داده‌ی تعریف شده دارد که استفاده نمی‌شود و هدف ویژه‌ای چون مشخص کردن آدرس گیرنده و فرستنده، تعریف نوع سرویس checksum و پروتکل‌های لایه‌های بالاتر دارد. اما فیلد جالب برای کار ما فیلد ۸ بیتی یا یک بایتی زمان TTL است. با در نظر گرفتن واقعیت پروتکل IP که بهترین عملکرد موجود را ارائه می‌دهد، بدین معنی که بهترین کار ممکن را برای تحویل بسته به مقصد نهایی انجام می‌دهد. بنابراین سازوکار ویژه‌ای باید وجود داشته باشد تا به زندگی بسته‌ای که امکان تحویل به مقصد را ندارد پایان دهد. بسته‌ها از روتری به روتر دیگری منتقل می‌شوند تا راهی به مقصد نهایی پیدا کنند و در هر بار انتقال از مقدار فیلد TTL یکی کم می‌شود تا زمانی که به صفر برسد، بسته به دور انداخته می‌شود و این کاهش سبب می‌شود که بسته‌ای را که نمی‌توان به مقصد تحویل داد، تا ابد در حلقه نیفتد و پهنای باند



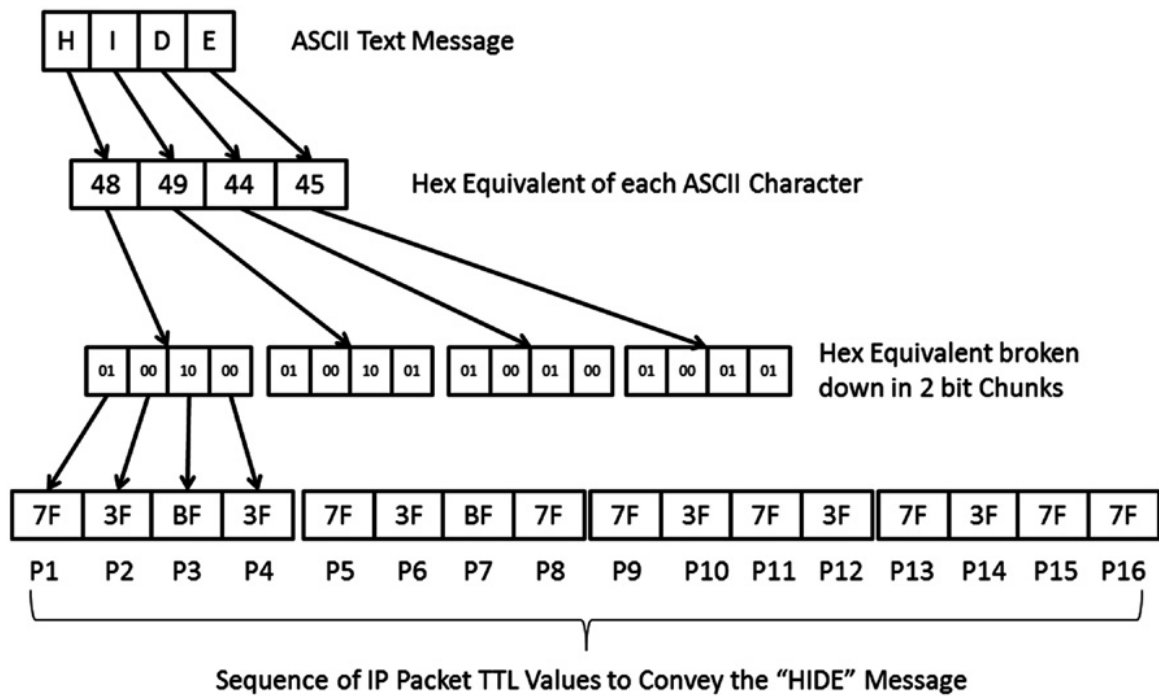
شبکه را اشغال نکند. نظر به ماهیت هوشمند مسیریابی مدرن در شبکه‌ی امروز، بیشتر بسته‌ها با چند گام به مقصدشان می‌رسند. بنابراین به نظر نمی‌رسد میانگین بالایی، دست‌کم از دیدگاه آماری در تحویل یک بسته IP وجود داشته باشد. تعداد گام‌هایی که معمولاً انتظار آن را داریم بین ۸ تا ۱۵ روتر است و بسیار کمتر از ۲۵۵ است که TTL آن را در نظر گرفته است. بنابراین به ندرت مقدار TTL پیش از رسیدن به مقصد به پایان می‌رسد مگر اینکه مقصد قابل دسترسی نباشد، بنابراین می‌توانیم دو بیت بارزش TTL را برای پنهان‌سازی داده‌ها بدون تأثیر بر تحویل آن‌ها به مقصد اختصاص دهیم. در شکل ۹-۱۰ دیاگرام بسته IP و فیلدهای مربوطه را نمایش داده و مقدار پیش فرض فیلد TTL برای تمام بسته‌ها برابر مقدار F۳ در مبنای ۱۶ (۶۳ روتر به صورت کلی) است و ۲ بیت بارزش برای جاسازی داده‌ها در هر بسته قابل استفاده است.



شکل ۹-۱۰: استفاده از دو بیت بارزش فیلد TTL برای پنهان‌سازی داده‌ها

در شکل ۹-۱۱ مثالی را از چگونگی عملکرد این روش پنهان‌سازی آورده‌ایم. اگر بخواهیم کلمه‌ی **HIDE** را در جریان بسته‌های IP جاسازی کنیم، نخست کد اسکی حروف را به مبنای ۱۶ تبدیل می‌کنیم، سپس مقدار حاصل را دو بیت دو بیت تقسیم و هر دو بیت را با دو بیت بارزش فیلد TTL یک بسته IP جایگزین می‌کنیم. ۶ بیت باقیمانده TTL همیشه برابر F۳ یا ۱۱۱۱۱۱x است پس مقدار **XX** با دو بیت جایگزین می‌شود. با جاسازی این مقادیر و ارسال ۱۶ بسته‌ی متوالی IP می‌توانیم پیام **HIDE** را ارسال کنیم. شاید ارسال فایلی به اندازه‌ی یک مگابایت با این روش دور از ذهن برسد، زیرا نیاز

به ارسال تقریباً چهار میلیون بسته دارد و  $1/4$  هر بایت در هر بسته IP متوالی ارسال می‌شود. با این حال اگر تعداد بسته‌های مورد نیاز برای پخش موسیقی یا ویدیو در یک ساعت را در نظر بگیرید به تعداد بسته‌هایی که برای ارسال این فایل در اختیار دارید پی می‌برید.



شکل ۹-۱۱: چگونگی ارسال پیام پنهانی در فیلد TTL

## کشف پنهان‌سازی داده‌ها در پروتکل شبکه

از دیدگاه تحلیلی، ما به نرم‌افزار تحلیل‌گر پروتکل یا پویشگر شبکه نیاز داریم. Wireshark ابزاری عالی برای بررسی و نمایش جزئیات پروتکل‌هاست؛ پس هرگاه به رفتار در شبکه مشکوک شدید، می‌توانید با استفاده از Wireshark، بسته‌های ارسالی و دریافتی بین دو رایانه را رصد کنیم. در این موقع تحلیل جزئیات هدر و بخش داده‌ی هر بسته RTP برای کشف مقادیر جاسازی شده غیرممکن و غیرعملی است. ولی از آنجایی که Wireshark امکان جستجو و فیلترینگ قوی دارد و هدر بسته‌ها معمولاً قالب مشخص دارند، تشخیص بسته‌های مشکل‌دار و ناقلا، کوشش ویژه‌ای را می‌طلبد.

وقتی می‌خواهیم بسته‌ای را که با روش ارسال، با تأخیر فرستاده شده را تشخیص دهیم، می‌بایست به شماره‌ی سریال بسته تأخیری دقت کنیم. مرتب کردن بسته‌ها بر اساس برچسب زمانی و با کمک گرفتن از زبان برنامه‌نویسی Python شناسایی بسته‌ای با تأخیر بی‌تی از چند ثانیه امکان‌پذیر است. کاملاً

به شما بستگی دارد که به ابزارهای تشخیص مهاجم نظیر SNORT متکی باشید یا بر اساس قواعد مدنظر خود، رفتار شبکه را به طور مداوم پیشگیری کنید.

## چکیده

همان‌گونه که پی بردید، راه و روش جاسازی اطلاعات پنهان در پروتکل‌های شبکه وجود داشته و با وجود میلیاردها پیام درخواست ارسال صفحات وب، ارتباطات VOIP، درخواست پخش موسیقی و ویدیو که تولید چند ده میلیارد بسته در روز در اینترنت می‌کنند، به سادگی امکان پنهان سازی پیام‌های کوچک و حتی مقدار زیاد اطلاعات را می‌دهند.

با ترکیب این روش با Zom، Trj، Keyloy و سایر کدهای مخرب که به وسیله‌ی کرم‌ها و ویروس‌ها به جا گذاشته می‌شوند، پرسش اینجاست که همین حالا چقدر و چگونه اطلاعات از سازمان شما نشت می‌کنند؟

~~~~~

### تحقیقات قضایی و راه گریز از آن

مدت‌های مدیدی شایع است که القاعده روش‌های پنهان‌سازی داده‌ها را برای مبادله‌ی اسناد مربوط به طرح‌های تروریستی به کار می‌گیرد. بیش از ده سال است که این سازمان از شیوه‌های استتار برای برقراری ارتباطات پوشیده‌ی خود استفاده می‌کند. ۱۶ می ۲۰۱۱ میلادی شخص اتریشی به نام Maqsood Lodin در برلین آلمان مورد بازجویی قرار گرفت. زیرا در لباس زیر خود یک کارت حافظه و ابزار ذخیره‌سازی دیجیتال پنهان کرده بود. کارت حافظه محتوا فایلهایی از جمله فایلهای تصویری بود. با بررسی بیشتر، پلیس آلمان بیش از ۱۰۰ فایل را کشف کرد که با روش‌های استتار در فایل تصویر پنهان شده بود و این فایل‌ها به وسیله‌ی گذرواژه هم محافظت می‌شد. پس از شکستن گذرواژه و دسترسی به محتوای فایل‌ها، مشخص شد که این فایل‌ها حاوی دستورالعمل‌های آموزش تروریستی، برنامه حرکت کشتی‌ها و زمان‌بندی حملات در اروپا می‌شد.

در جهان دیجیتال امروزی، پنهان‌سازی داده‌ها، خود را برای استفاده از امکانات ارتباطات دیجیتال تحت پوشش با هدف مشترک گریز از کشف شدن، بازسازی کرده است. با دانستن روش‌های به‌کاررفته در بازرسی‌های قضایی<sup>۱</sup> برای پیشگیری از رو شدن دستتان می‌بایست از روش‌های ضعیف استتار دوری کنید. با این دانسته‌ها، شالوده‌ی لازم را برای استفاده از روش‌های بهینه‌شده‌ی پوشیده‌نگاری، برای بیشتر موقعیت‌های بحرانی را در اختیار دارید. در این فصل روش‌های بیشتری از بازرسی قضایی و روش‌های گریز از آن را بررسی می‌کنیم که در پنهان‌سازی داده‌ها کاربرد داشته و در فصل‌های پیشین به آن اشاره نکرده‌ایم.

---

۱ Anti-Forensic

## راه‌های گریز از بازرسی‌های قضایی: روش‌های کار خود را پنهان کنید

یکی از اشتباه‌های رایجی که بیشتر کاربران روش‌های پنهان‌سازی داده‌ها مرتکب می‌شوند، باقی گذاشتن نه تنها پیامی است که در فایل حامل پنهان کرده‌اند، بلکه خود فایل حامل اصلی را نیز نگه می‌دارند. همان‌گونه که در فصل‌های پیشین شرح دادیم، روش‌های زیادی برای تشخیص تفاوت بین دو فایل به ظاهر یکسان وجود دارد. در نتیجه قویاً توصیه می‌شود، فایلی را که به عنوان فایل حامل و با تغییر داده‌های آن پنهان‌سازی را انجام داده‌اید را حتماً حذف نمایید.

مدارک مربوط به نرم‌افزار پنهان‌سازی داده‌ها هم باید از روی رایانه پاک شود. اگر برنامه‌ی پوشیده‌نگاری بر روی رایانه شخصی مزنون پیدا شود، به این معنی است که فایل‌های حامل بسیاری هم بر روی همان رایانه وجود دارد. اگر این فایل‌های حامل پیدا شوند، تنها مانع باقیمانده بین بازرسان در آشکار کردن داده‌های پنهان فقط گذرواژه است. اگر این گذرواژه در جای دیگر مثل Login به رایانه هم مورد استفاده قرار گرفته باشد، بازرسان یک گام دیگر به آشکارسازی داده‌های پنهان نزدیک می‌شوند. برخی کاربران پس از استفاده از نرم‌افزار پنهان‌سازی، آن را حذف می‌کنند و ردپاهای به‌جامانده از نصب نرم‌افزار و محتویات و Recycle Bin را معمولاً فراموش می‌کنند.

وقتی می‌خواهید فایلی را در عکس دیجیتال پنهان کنید، پیشنهاد می‌شود که از عکس‌های موجود و در دسترس استفاده نکنید و خود اقدام به گرفتن عکس جدید نمایید. هرگز از عکس‌های موجود برای پنهان‌سازی استفاده نکنید. یک فایل عکس معمولی به بازرسان امکان می‌دهد که فایل اصلی را در محل دیگری غیر از رایانه مزنون یافته و با مقایسه فایل اصلی و فایل‌های موجود در رایانه اقدام به مقایسه تفاوت‌ها و کشف داده‌های پنهان نمایند. مثلاً بانک اطلاعاتی وجود دارد که توسط آژانس‌ها و سرویس دهندگان خاصی نگهداری می‌شود. با «هش» داده‌های عکس مزنون و مقایسه آن با هش شناخته شده از آن فایل، فایلی که متفاوت است فوراً مشخص می‌شود.

نگهداری نرم‌افزار پوشیده‌نگاری و فایل‌های حامل مورد استفاده بر روی ابزارهای ذخیره‌سازی قابل حمل، گرچه نخستین گام در حذف مدارک در تحقیقات دال بر استفاده از روش‌های پنهان‌سازی است ولی کافی نیست، پس اجازه دهید با روش‌های کارا در بازرسی‌های قضایی کار را ادامه دهیم.

## نقش گذرواژه در پنهان‌سازی داده

گذرواژه، همواره موضوع در خور توجهی بوده است. همیشه به یاد داشته باشید وقتی که پیامی را در فایل حامل پنهان می‌کنید، از گذرواژه‌ی پیچیده استفاده کنید. ویژگی گذرواژه شامل موارد زیر است:

(۱) از گذرواژه‌ی متفاوت از گذرواژه‌ی سیستم‌عامل، گذرواژه‌ی مرورگرهای اینترنت و تجهیزات شبکه استفاده نمایید.

(۲) از ترکیب کلمات کوچک و بزرگ و علامت‌های ویژه‌ی نگارش و اعداد استفاده کنید.

(۳) اگر نیاز است که کلمه عبور را در محل دیگری ذخیره کنید از سایت <http://passwordsafe.sourceforge.net> کمک بگیرید.

بسیاری از برنامه‌های پنهان‌سازی نیاز به تعیین گذرواژه به وسیله‌ی کاربر دارند. وقتی که داده‌ای را پنهان می‌کنید، پیشنهاد می‌شود از گذرواژه‌ی پیچیده استفاده کنید. در طول تدریس از تعداد زیادی مدیران سیستم که از وجود حروف پنهان در صفحه کلید که می‌توان از آن‌ها در گذرواژه استفاده کرد بی‌اطلاع بودند، متعجب شدم. بسیاری از حملات از نوع دیکشنری و روش‌های تولید تمام حالات حروف، این علائم را در نظر نمی‌گیرند. نمونه‌هایی از این علائم به شکل زیر است:

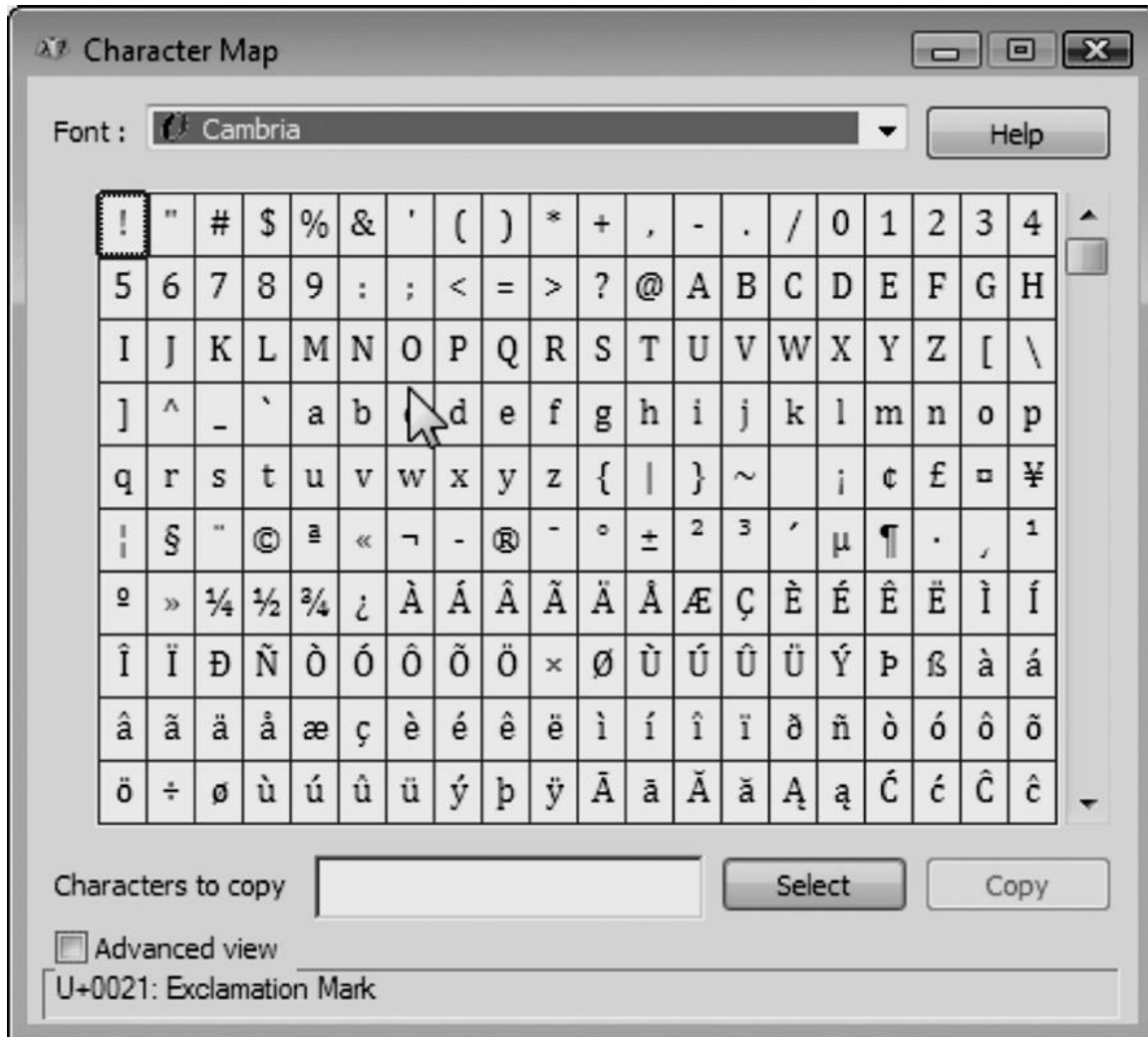
© CTRL] + [ALT] + [C] gives]

® CTRL] + [ALT] + [R] gives]

™ CTRL] + [ALT] + [T] gives]

€ CTRL] + [ALT] + [E] gives]

شکل ۱۰-۱، لیست مفصلی از علائم ویژه در ویندوز را نشان می‌دهد.



شکل ۱۰-۱: لیست علائم و نویسه های ویژه

استفاده از یکی یا بیشتر از این علائم ویژه در گذرواژه سبب می شود تا گذرواژه به وسیله ی بسیاری از برنامه های شکستن گذرواژه کشف شود. استفاده از این علائم ویژه نه تنها در گذرواژه ی مورد استفاده در پنهان سازی داده ها پسندیده است، بلکه در مسائل پیرامون امنیت رایانه به طور کلی توصیه می شود. به علاوه با افزودن تعداد علائم ویژه در گذرواژه، کار شکستن آن را به طور چشم گیری مشکل تر می کنید. این روش ها مانع می شود که بازرسان به وسیله ی گذرواژه ی خودتان داده های پنهانان را مشاهده نمایند.



## روش‌های تان را پنهان نمایید

در سیستم عامل ویندوز، می‌توانید از نرم‌افزار `cleanmgr` برای پاک‌سازی سیستم از هر نوع مدرک دال بر استفاده از نرم‌افزارهای پنهان‌سازی داده‌ها استفاده کنید. البته این راه حل کامل نیست، ولی یک روش سریع برای پاک‌سازی رایانه است. برای اجرای آن کافی است در خط فرمان دستور زیر را تایپ کنید:

`c:\cleanmgr`

در زمان اجرا، به کاربر پیام می‌دهد که درایو مورد نظر را انتخاب نماید، سپس موارد زیر را از روی

سیستم پاک می‌کند:

- فایل‌های موقت اینترنتی
- فایل‌های موقت نصب نرم‌افزار
- فایل‌های موقت `offline`
- فایل‌های برنامه‌های دانلود شده
- `Recycle Bin` فایل‌های موقت سیستم عامل ویندوز
- اجزاء بهینه‌ساز ویندوز که دیگر استفاده نمی‌شود.
- فایل‌های قدیمی حاصل از اجرای دستور `chkdsk`
- فایل‌های کاتالوگ برای اندیکس سازی محتوا

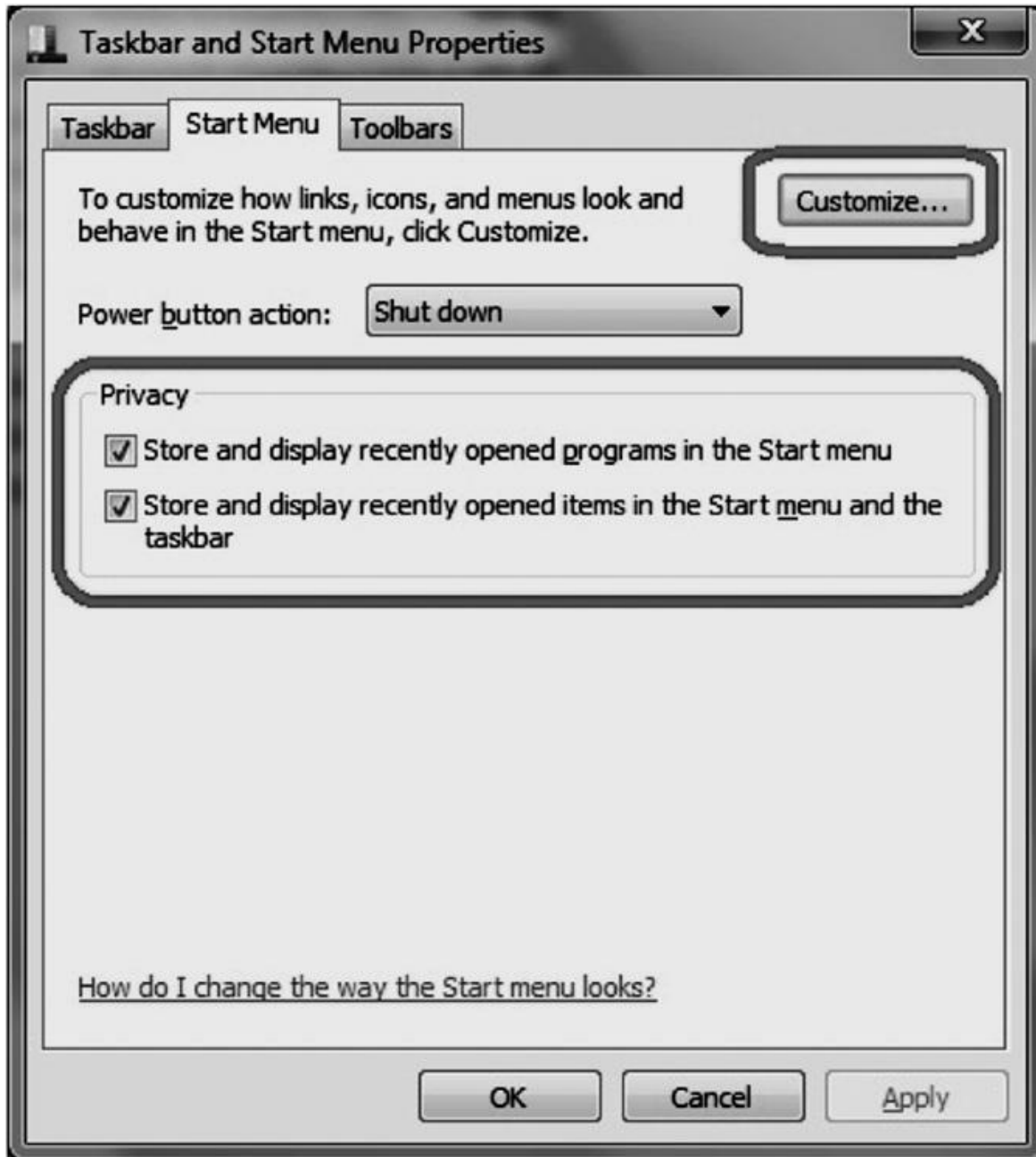
سیستم عامل ویندوز، فرکانس اجرا را پیگیری کرده و برنامه‌های پرکاربرد را در منوی شروع قرار می‌دهد. برای پاک کردن لیست برنامه‌های استفاده شده در ویندوز ۷ بر روی گزینه `Start` کلیک راست زده و بر روی گزینه `property` کلیک کنید، سپس گزینه‌های زیر `privacy` را از حالت انتخاب خارج کنید (شکل ۱۰-۲). این گزینه‌ها شامل

Store and display recently opened programs in the Start menu

و

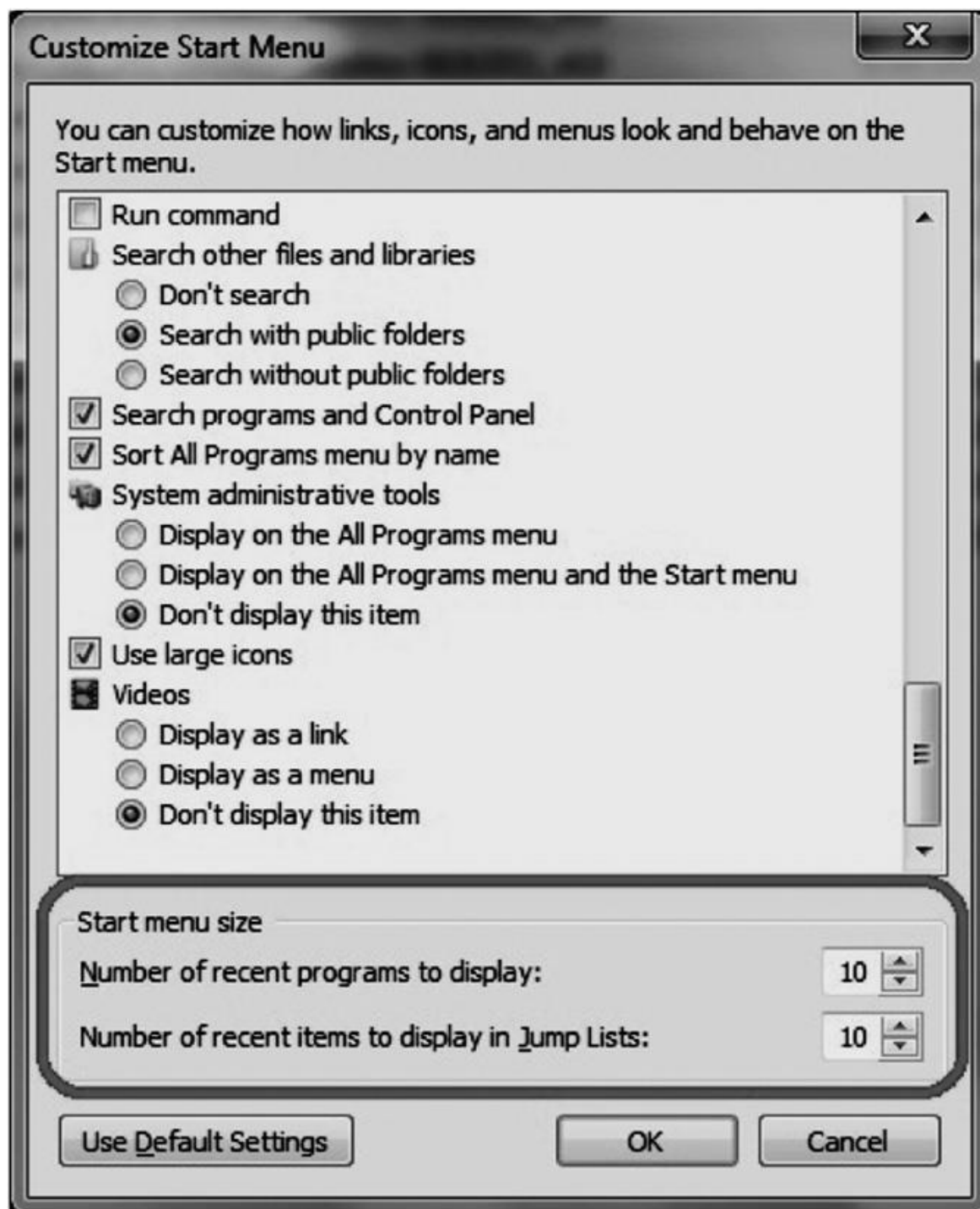
Store and display recently opened items in the Start menu and taskbar

است.



شکل ۱۰-۲: این گزینه‌ها را برای عدم نمایش برنامه‌هایی که اخیراً استفاده شده‌اند غیرفعال نمایید.

همچنین بر روی گزینه‌ی customize کلیک کرده و تعداد برنامه‌هایی که اخیراً استفاده شده‌اند را همانند شکل ۱۰-۳ به صفر تغییر دهید.



شکل ۱۰-۳: پیکربندی منوی شروع در ویندوز

این تنظیمات همچنین به شما اجازه می دهند که مدارک حاکی از آخرین برنامه ی اجرا شده را از بین ببرید.

## پیگرد قضایی

روش های گوناگونی وجود دارد که مشخص کنیم رایانه ای مظنون به اجرای برنامه های پنهان سازی داده - هاست، که شامل:

- ✓ نرم افزارهای پنهان سازی داده ها، نصب شده بر روی رایانه
- ✓ صفحات وب کش شده بر روی رایانه که بیانگر دسترسی به صفحات اینترنتی است که خدمات پنهان سازی داده ها را ارائه می دهند.
- ✓ عکس های کش شده که بیانگر دسترسی و دانلود احتمالی نرم افزارهای پنهان سازی داده هاست.
- ✓ سایر ردپاها که بیانگر نصب و استفاده از نرم افزارهای پنهان سازی داده ها در گذشته بوده است که شامل ریجستری، فایل های به جامانده پس از حذف نرم افزار پنهان سازی و فایل های Thumb می شود.

## جستجوی نرم افزارهای پنهان سازی داده ها در رایانه

از دیدگاه بیشتر جرم، آشکار است که با نگاه گذرا به سیستم مظنون، استفاده از نرم افزارهای پنهان سازی داده ها را می توان مشاهده کرد. پس برای این کار از مشاهده ی برنامه ی نصب شده ی فعلی گرفته تا جستجوی پوشه هایی که منتهی به آشکار شدن ردپا از نرم افزار نصب شده می شود می توان کمک گرفت. برای مثال در لینوکس Ubuntu نرم افزارهای نصب شده بر روی رایانه را با دستور زیر می توانید مشاهده نمایید:

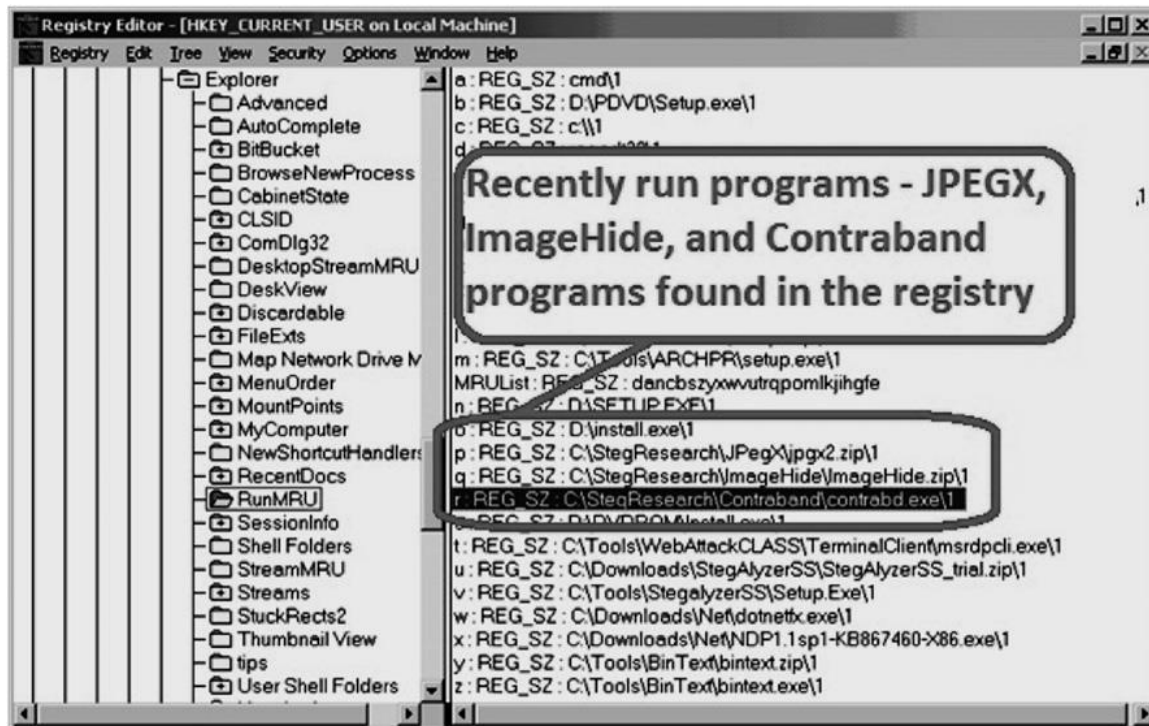
```
#sudo dpkg --get-selections > listofpkg
```

توجه کنید که برخی نرم افزارهای پنهان سازی داده به نصب نیاز ندارند؛ در نتیجه می توان آن ها را از روی CD، حافظه های جانبی و... اجرا کرد. برای مشاهده ی برنامه هایی که اخیراً در ویندوز اجرا شده اند، regedit را اجرا و محتویات کلید زیر را مشاهده نمایید.

User Key:

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\
Explorer\RunMRU]
```

شکل ۱۰-۴، لیستی از برنامه‌هایی که اخیراً اجرا شده‌اند را در رجیستری ویندوز نشان می‌دهد. در این مثال با مشاهده‌ی این لیست، تعدادی از نرم‌افزارهای استتار از قبیل cona را مشاهده می‌کنید.



شکل ۱۰-۴: پیدا کردن برنامه‌ی پنهان‌سازی داده‌ها در رجیستری

روش‌های گوناگون و راه‌های دیگری وجود دارند که می‌توان رد بجا مانده از نرم‌افزارهای نصب شده پنهان‌سازی داده‌های قبلی را پیدا کرد؛ حتی برخی از آن‌ها به صورت خودکار این‌گونه تحلیل‌ها را انجام می‌دهند. تعدادی از ابزارهای رایج در این خصوص را باهم مرور می‌کنیم.

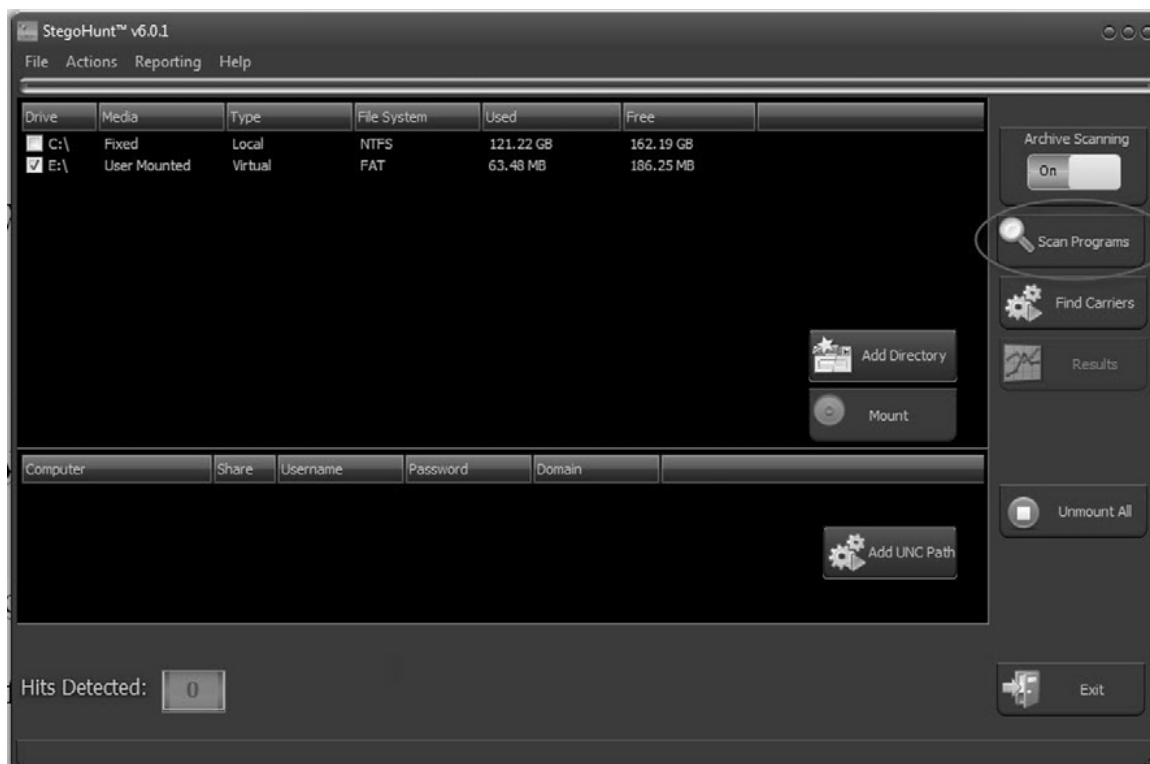
## پیدا کردن ردپاها

بین کاربران باتجربه مرسوم است که نرم‌افزار پنهان‌سازی داده را پس از اجرا حذف کرده یا آن‌ها را از روی حافظه‌های قابل حمل اجرا کنند. هر دو روش مدارک بی‌نهایت مفیدی را برای بازرسان در خلال تحقیقات به جا می‌گذارد. هر ردپایی از داده‌های رجیستری گرفته تا پوشه‌های موقت نصب نرم‌افزار، راهی را برای بازرسان در خلال تحقیق از رایانه مظنون می‌گشایند.

ولی راه دیگری برای مشخص کردن وجود نرم‌افزار پوشیده‌نگاری در حال حاضر یا در گذشته در رایانه مظنون وجود دارد. سازمان‌های گوناگونی مثل وزارت دفاع و NIST فایل‌های «هش» با پسوند dll را در اختیار دارند که معرف فایل‌هایی است در خلال نصب نرم‌افزارهای پنهان‌سازی در رایانه ایجاد می‌شوند. اکنون نرم‌افزارهایی وجود دارند که به بازرسان امکان می‌دهد رایانه مظنون را در خصوص وجود این‌گونه فایل‌ها بررسی کرده و با مقایسه فایل‌های هش به وجود نرم‌افزار پنهان‌سازی داده‌ها پی ببرند. این نرم‌افزارها گاهی در ریجستری به دنبال ردپا می‌گردند. اگرچه نرم‌افزار مذکور از روی رایانه پاک شده باشد.

### **TMWetStone Technologies StegoHunt**

این نرم‌افزار به بازرسان امکان می‌دهد که به دنبال برنامه‌های شناخته شده در حوزه استتار و پنهان‌سازی داده‌ها گشته و فایل حامل (از نوع صوت، تصویر، عکس، متن) که حاوی داده‌های پنهان هستند را مشخص کنند. WetStone در طول بیش از یک دهه، مجموعه‌ای بالغ بر دو هزار نرم‌افزار پنهان‌سازی داده‌ها را گردآوری کرده است. StegoHunt از ترکیبی از اعداد اولویت‌دار فیبوناچی و امضاء دیجیتالی SHA و MD5 برای تشخیص دقیق نرم‌افزارهای پنهان‌سازی داده‌ها استفاده می‌کند. این نرم‌افزار قادر به اسکن درایو، پوشه‌ها، عکس‌ها و رایانه‌های متصل به شبکه، برای تشخیص چنین حملاتی است (شکل ۱۰-۵).



شکل ۱۰-۵: صفحه انتخاب برنامه‌ی پویش StegoHunt

پس از به پایان رسیدن اسکن، StegoHunt نتایج را با جزئیات به شکل جایی یا شکلی که در تصویر ۱۰-۶ مشاهده می‌کنید نمایش می‌دهد. این جزئیات شامل اطلاعاتی در خصوص فایل با داده‌های پنهان به همراه برنامه‌ی پنهان‌سازی یا استتار به‌کاررفته را نشان می‌دهد، StegoHunt همچنین زمان دسترسی، زمان تغییر در فایل و زمان ایجاد فایل، فایل «هش» مربوطه، نام فایل، پوشه، نام اصلی فایل که توسط نرم‌افزار پنهان‌سازی داده‌ها مورد استناد قرار گرفته را هم نشان می‌دهد. (شایان یادآوری است که برخی از بدافزارها از تغییر نام یا پسوند فایل برای پنهان کردن فعالیت‌هایشان کمک می‌گیرند).

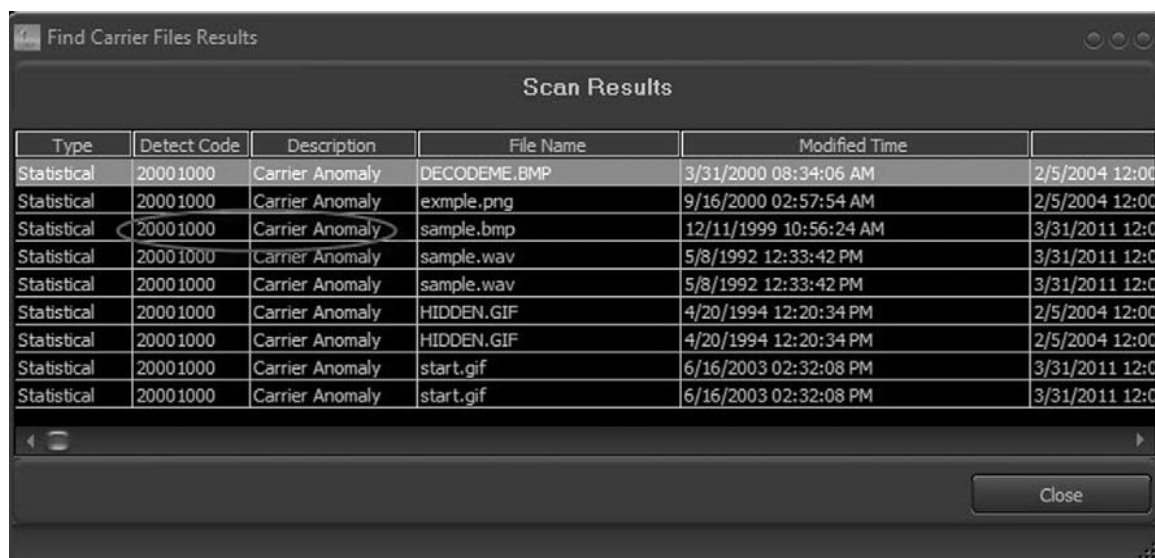
| Category      | Program                      | File Name             | Modified Time          |
|---------------|------------------------------|-----------------------|------------------------|
| Steganography | InvisibleSecrets             | blowfish.dll          | 12/18/1999 11:32:44 AM |
| Steganography | InvisibleSecrets             | blowfish.dll          | 1/18/2001 09:16:08 PM  |
| Steganography | InvisibleSecrets2002         | blowfish.dll          | 7/11/2001 08:10:46 AM  |
| Steganography | BMP-Embedding                | bmp-embed.exe         | 9/16/2000 02:33:44 AM  |
| Steganography | InvisibleSecrets2002         | bmpcarrier.dll        | 1/29/2001 11:12:34 PM  |
| Steganography | InvisibleSecrets             | bmpcarrier.dll        | 12/18/1999 11:09:40 AM |
| Steganography | InvisibleSecrets2002         | bmpcarrier.dll        | 1/29/2001 11:12:34 PM  |
| Steganography | BlindSide                    | BSIDE.EXE             | 4/29/2000 11:04:18 AM  |
| Steganography | BlindSide                    | BSIDE.EXE             | 4/29/2000 11:04:18 AM  |
| Steganography | Scytale                      | BUSEINE.ASC           | 8/22/2002 04:08:48 PM  |
| Steganography | Scytale                      | BUSEINE.ASC           | 8/22/2002 04:06:14 PM  |
| Steganography | InvisibleSecrets             | buynow.rtf            | 2/4/2001 10:45:36 PM   |
| Steganography | InvisibleSecrets2002         | buynow.rtf            | 2/2/2002 11:52:26 AM   |
| Steganography | InvisibleSecrets2002         | cast128.dll           | 1/18/2001 09:16:22 PM  |
| Steganography | InvisibleSecrets2002         | cast128.dll           | 1/18/2001 09:16:22 PM  |
| Steganography | Scytale                      | COMPUTER.DCT          | 1/1/1999 12:50:00 AM   |
| Steganography | Scytale                      | COMPUTER.DCT          | 1/1/1999 12:50:00 AM   |
| Steganography | BlindSide                    | Copy (2) of BSIDE.EXE | 4/29/2000 11:04:18 AM  |
| Steganography | BlindSide                    | Copy of BSIDE.EXE     | 4/29/2000 11:04:18 AM  |
| Steganography | HideInPicture-SourceCodeInfo | COPYING               | 9/10/1999 09:53:24 PM  |
| Steganography | InvisibleSecrets             | default1.html         | 1/30/2000 10:42:32 AM  |
| Steganography | S-Tools                      | DES.DLL               | 4/18/1994 01:56:12 PM  |
| Steganography | S-Tools                      | DES.DLL               | 4/18/1994 01:56:12 PM  |
| Steganography | InvisibleSecrets2002         | diamond2.dll          | 1/18/2001 10:24:40 PM  |
| Steganography | InvisibleSecrets2002         | diamond2.dll          | 1/18/2001 10:24:40 PM  |

شکل ۱۰-۶: گزارش حاصل از اجرای نرم افزار StegoHunt

همچنین StegoHunt توانایی اسکن کردن فایل های حامل (فایل هایی که حاوی داده های پنهان هستند) را دارد. برای این منظور تابعی را فراخوانی می کند تا فایل های عکس، فیلم، صوت، ویدئو و متن را جستجو کند. الگوریتم بکار رفته در آن بسیار دقیق بوده و توانایی تشخیص ناموزونی<sup>۱</sup> جزئی در فایل های عکس یا چندرسانه ای بوده که در حین پنهان سازی داده ها به وجود می آید. به علاوه این تابع امضاء و ساختارهای ساختگی حاصل از فعالیت پنهان سازی داده را تشخیص می دهد. مشاهده ی نتایج پوشش، خلاصه ای از محل دقیق فایل های مظنون به داشتن داده های پنهان را مشخص می کند. به علاوه هر فایل، گروه و کد تشخیصی دارد که اطلاعات بیشتری از یافته ها را نشان می دهد (شکل ۱۰-۷).

<sup>1</sup> Anomaly



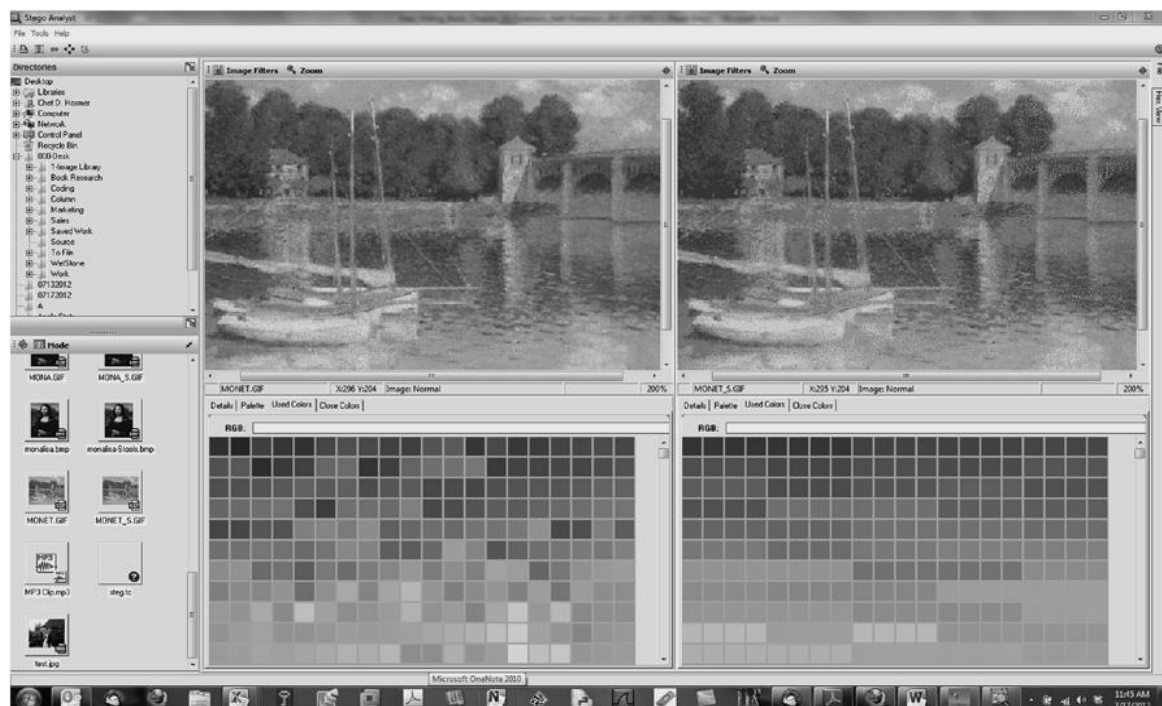


| Type        | Detect Code | Description     | File Name    | Modified Time          |
|-------------|-------------|-----------------|--------------|------------------------|
| Statistical | 20001000    | Carrier Anomaly | DECODEME.BMP | 3/31/2000 08:34:06 AM  |
| Statistical | 20001000    | Carrier Anomaly | exmple.png   | 9/16/2000 02:57:54 AM  |
| Statistical | 20001000    | Carrier Anomaly | sample.bmp   | 12/11/1999 10:56:24 AM |
| Statistical | 20001000    | Carrier Anomaly | sample.wav   | 5/8/1992 12:33:42 PM   |
| Statistical | 20001000    | Carrier Anomaly | sample.wav   | 5/8/1992 12:33:42 PM   |
| Statistical | 20001000    | Carrier Anomaly | HIDDEN.GIF   | 4/20/1994 12:20:34 PM  |
| Statistical | 20001000    | Carrier Anomaly | HIDDEN.GIF   | 4/20/1994 12:20:34 PM  |
| Statistical | 20001000    | Carrier Anomaly | start.gif    | 6/16/2003 02:32:08 PM  |
| Statistical | 20001000    | Carrier Anomaly | start.gif    | 6/16/2003 02:32:08 PM  |

شکل ۱۰-۷: گزارش نرم افزار StegoHunt و فایل های مظنون، داشتن داده های پنهان

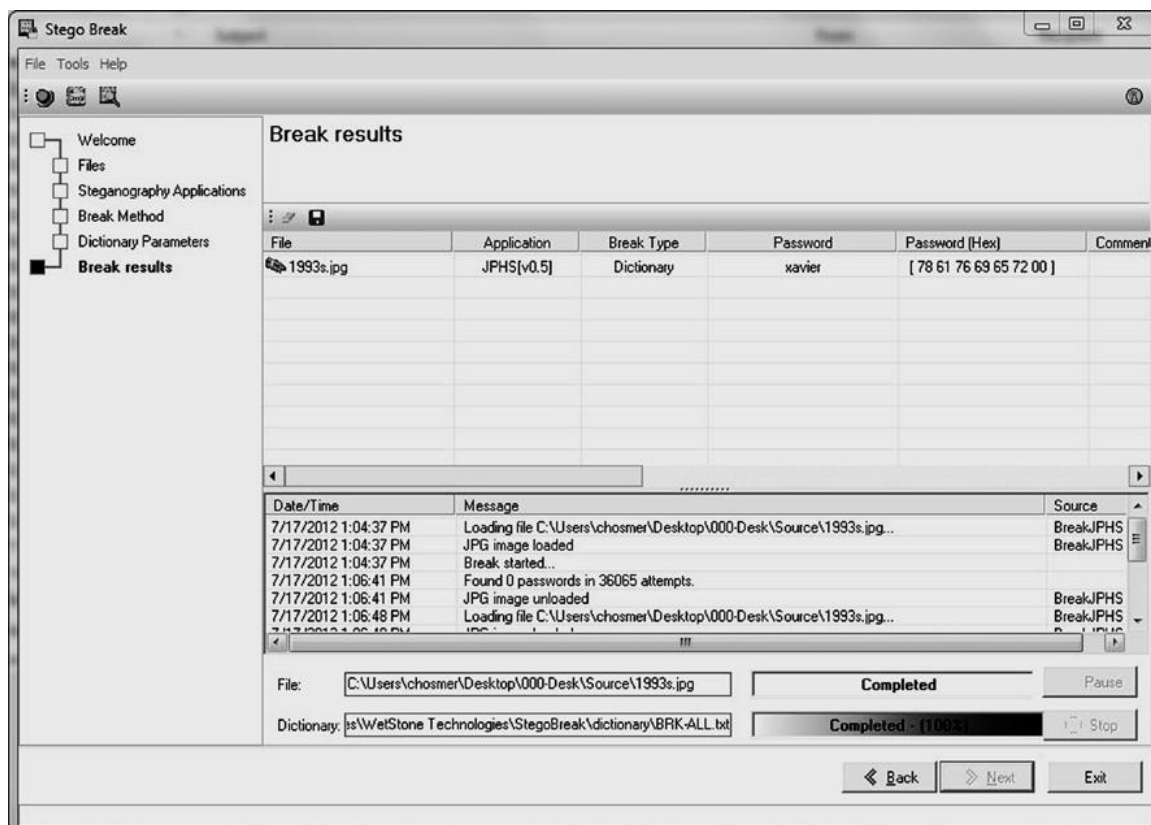
وقتی که فایل مظنون به داشتن داده های پنهان شده یافت شد، StegoHunt بررسی عمیق و ژرفی را شروع کرده و اقدام به عمل crack (شکستن) فایل حامل برای آشکارسازی داده های پنهان شده در آن می کند.

تحلیل ژرف تر، اطلاعات دقیق و جزئی از ویژگی های دیداری و نوشتاری فایل های عکس، صوت، تصویر و متن را ارائه می دهد و ابزارهای بسیاری برای تحلیل فعالیت های پنهان سازی داده ها در این گونه فایل ها در اختیار شما قرار می دهد (شکل ۱۰-۸).



شکل ۱۰-۸: بررسی وجود داده‌های پنهان در فایل عکس به وسیله‌ی نرم‌افزار StegoBreak

StegoBreak از روش دیکشنری یا آزمودن تمام حالت‌های کلید برای شکستن رمز فایل حامل استفاده می‌کند و در صورت موفقیت، گذرواژه‌ای که نرم‌افزار استتار از آن استفاده کرده را بر می‌گرداند؛ سپس به وسیله‌ی روش‌های شناخته شده‌ی استتار یا پنهان‌سازی داده‌ها، و با استفاده از رمز کشف شده، اقدام به آشکار کردن داده‌های پنهان می‌نماید (شکل ۱۰-۹).



شکل ۱۰-۹: پیدا کردن گذرواژه‌ی فایل‌هایی که داده‌های پنهان‌شان را به وسیله‌ی آن حفاظت می‌کنند.

## تشخیص و مشاهده‌ی تصاویر کش شده (ابزارهای حسابرسی کش)

علاوه بر پویش رایانه برای پیدا کردن خود نرم‌افزار پنهان‌سازی، بازرسان می‌توانند استفاده از سایت‌هایی که خدماتی در خصوص پنهان‌سازی داده انجام می‌دهند را نیز مد نظر داشته باشند. رایانه مظنون که حاوی اطلاعات بسیاری در مورد سایت‌های بازدید شده و عادت‌های سیر و سیاحت در اینترنت است؛ کاملاً محتمل است که از رایانه‌ای که نرم‌افزار پنهان‌سازی داده‌ها را دانلود کرده، نیاز هم استفاده کرده باشد. تحلیل آدرس URL بازدید شده و تصاویر کش شده می‌تواند راه عالی برای مشخص کردن این موضوع باشد که آیا رایانه نیاز به بررسی دقیق‌تر دارد یا خیر؟ همچنین مهم است که آیا مظنون عضو گروه گپ خاصی است و این گروه هم ارتباطی به تحقیقات دارند یا خیر.

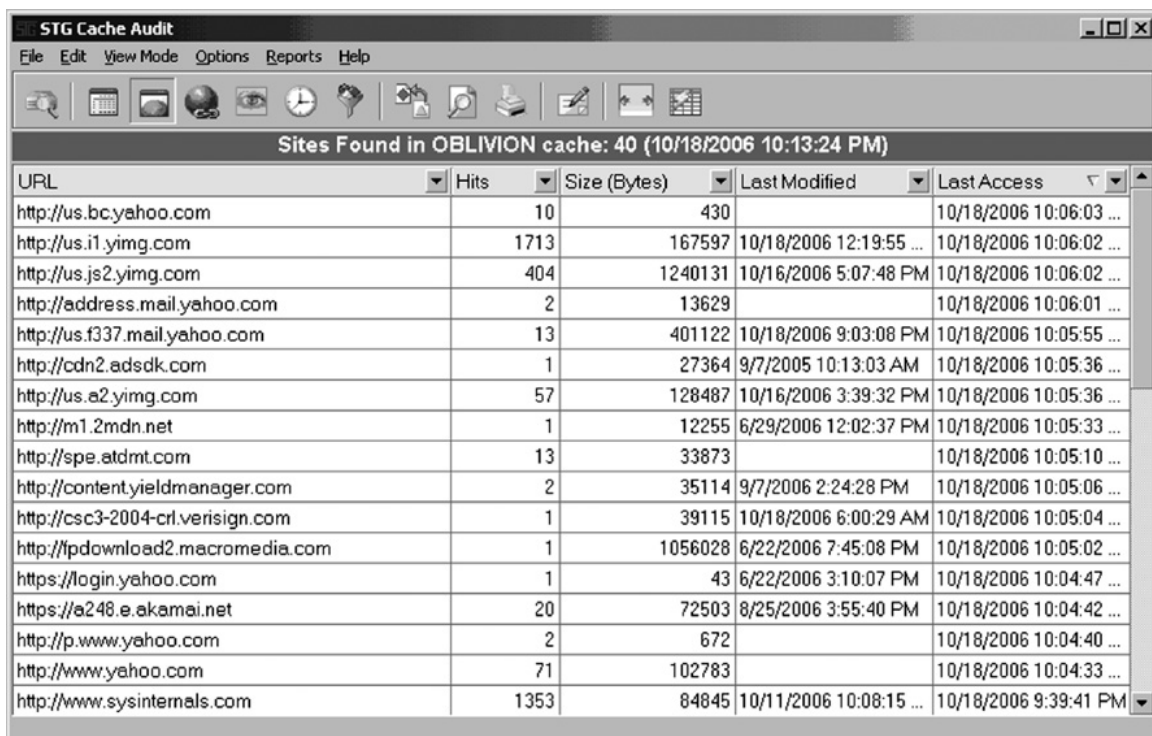
در تحقیقات قضایی لازم است برای بررسی عکس‌های موجود در رایانه، به واری‌های عکس‌های کش شده هم بپردازید. اگر فرد مظنون اقدام به حذف نرم‌افزار پنهان‌سازی داده و فایل حامل نماید، پوشه حافظه پنهان هنوز محلی برای آشکارسازی عکس‌ها است و این دلیل دیگری مبنی بر استفاده از سایت-هایی است که خدمات پنهان‌سازی داده‌ها را ارائه می‌دهند. در موارد وحشت‌افکنی، عکس‌های cach

شده، تصاویری را آشکار می‌کنند که ممکن است از سایتی با اهداف وحشت‌افکنی برای دانلود در حوزه‌ی دیگری ارسال شده و تحلیل این عکس‌ها ممکن است وجود داده‌های پنهان شده و گاهی خود داده‌های پنهان را آشکار نماید.

## نرم‌افزار حسابرسی STG Cache Audit

STG Cache Audit، قابل دریافت از نشانی

[www.snapfiles.com/screenshots/stgcache.htm](http://www.snapfiles.com/screenshots/stgcache.htm) نرم‌افزار تحت ویندوز است که امکان حسابرسی از کش، کوکی و... تاریخچه سیستم را فراهم کرده و عادت‌های گشتن در اینترنت در رایانه مظنون را بررسی می‌کند. می‌توانید کلماتی را به عنوان فیلتری تعیین کرده تا سایت‌های مرتبط با کلمه-ی کلیدی معینی را مشاهده نمایید. گزینه *site view* امکان مشاهده‌ی سایت‌های ملاقات شده و تعداد ملاقات‌ها را نمایش می‌دهد. در شکل ۱۰-۱۱ نتیجه‌ی اجرای STG بر روی رایانه مظنون را نشان می‌دهد و به سرعت می‌توان سایت‌های بازدید شده به وسیله‌ی مظنون را مشخص کرد.



The screenshot shows the STG Cache Audit application window. The title bar reads "STG Cache Audit". The menu bar includes "File", "Edit", "View Mode", "Options", "Reports", and "Help". Below the menu bar is a toolbar with various icons. The main window displays a table titled "Sites Found in OBLIVION cache: 40 (10/18/2006 10:13:24 PM)". The table has five columns: "URL", "Hits", "Size (Bytes)", "Last Modified", and "Last Access". The table lists various websites and their corresponding statistics.

| URL                               | Hits | Size (Bytes) | Last Modified           | Last Access             |
|-----------------------------------|------|--------------|-------------------------|-------------------------|
| http://us.bc.yahoo.com            | 10   | 430          |                         | 10/18/2006 10:06:03 ... |
| http://us.i1.yimg.com             | 1713 | 167597       | 10/18/2006 12:19:55 ... | 10/18/2006 10:06:02 ... |
| http://us.js2.yimg.com            | 404  | 1240131      | 10/16/2006 5:07:48 PM   | 10/18/2006 10:06:02 ... |
| http://address.mail.yahoo.com     | 2    | 13629        |                         | 10/18/2006 10:06:01 ... |
| http://us.f337.mail.yahoo.com     | 13   | 401122       | 10/18/2006 9:03:08 PM   | 10/18/2006 10:05:55 ... |
| http://cdn2.adsdk.com             | 1    | 27364        | 9/7/2005 10:13:03 AM    | 10/18/2006 10:05:36 ... |
| http://us.a2.yimg.com             | 57   | 128487       | 10/16/2006 3:39:32 PM   | 10/18/2006 10:05:36 ... |
| http://m1.2mdn.net                | 1    | 12255        | 6/29/2006 12:02:37 PM   | 10/18/2006 10:05:33 ... |
| http://spe.atdmt.com              | 13   | 33873        |                         | 10/18/2006 10:05:10 ... |
| http://content.yieldmanager.com   | 2    | 35114        | 9/7/2006 2:24:28 PM     | 10/18/2006 10:05:06 ... |
| http://csc3-2004-crl.verisign.com | 1    | 39115        | 10/18/2006 6:00:29 AM   | 10/18/2006 10:05:04 ... |
| http://pdownload2.macromedia.com  | 1    | 1056028      | 6/22/2006 7:45:08 PM    | 10/18/2006 10:05:02 ... |
| https://login.yahoo.com           | 1    | 43           | 6/22/2006 3:10:07 PM    | 10/18/2006 10:04:47 ... |
| https://a248.e.akamai.net         | 20   | 72503        | 8/25/2006 3:55:40 PM    | 10/18/2006 10:04:42 ... |
| http://p.www.yahoo.com            | 2    | 672          |                         | 10/18/2006 10:04:40 ... |
| http://www.yahoo.com              | 71   | 102783       |                         | 10/18/2006 10:04:33 ... |
| http://www.sysinternals.com       | 1353 | 84845        | 10/11/2006 10:08:15 ... | 10/18/2006 9:39:41 PM   |

شکل ۱۰-۱۰: مشاهده‌ی تاریخچه‌ی سایت‌های ملاقات شده در اینترنت به وسیله‌ی STG

گزینه‌ی Histoy view به کاربر امکان مشاهده‌ی وب سایت‌های بازدید شده را بر اساس زمان بازدید می‌دهد. بازرسان همچنین می‌توانند جزئیات بیشتری از نشانی URL خاصی همانند که در شکل ۱۰-۱۱ مشاهده می‌کنید را به دست آورند.

| Cache Summary for OBLIVION (10/18/2006 10:13:24 PM) |                |                           |
|---|----------------|---------------------------|
| Item Type   | Total          | Size                      |
| URL   | 454 (40 sites) | 4.13 MB (4,331,611 bytes) |
| Cookies   | 15             | 0.00 MB (4,025 bytes)     |
| History   | 1              | 0.00 MB (0 bytes)         |
| Totals:   | 470            | 4.13 MB (4,335,636 bytes) |
| Filtered  | 1              | 0.03 MB (32,223 bytes)    |

| Extra Information |   |
|-------------------|---|
| Item              | Path  |
| URL Path          | C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files |
| Cookies Path      | C:\Documents and Settings\Administrator\Cookies                                 |
| History Path      | C:\Documents and Settings\Administrator\Local Settings\History                  |
| Notes             |   |

شکل ۱۰-۱۱: مشاهده‌ی تاریخچه حسابرسی کش در STG

کاربر همچنین می‌تواند گزارش نرم‌افزار را به شکل فایل html، فایل متنی csv و فایل صفحه گسترده‌ی اکسل ذخیره کند تنها مدارک پنهان‌سازی داده‌ها در تمام طول تحقیقات است.

## مدارک موجود در Thumbnails

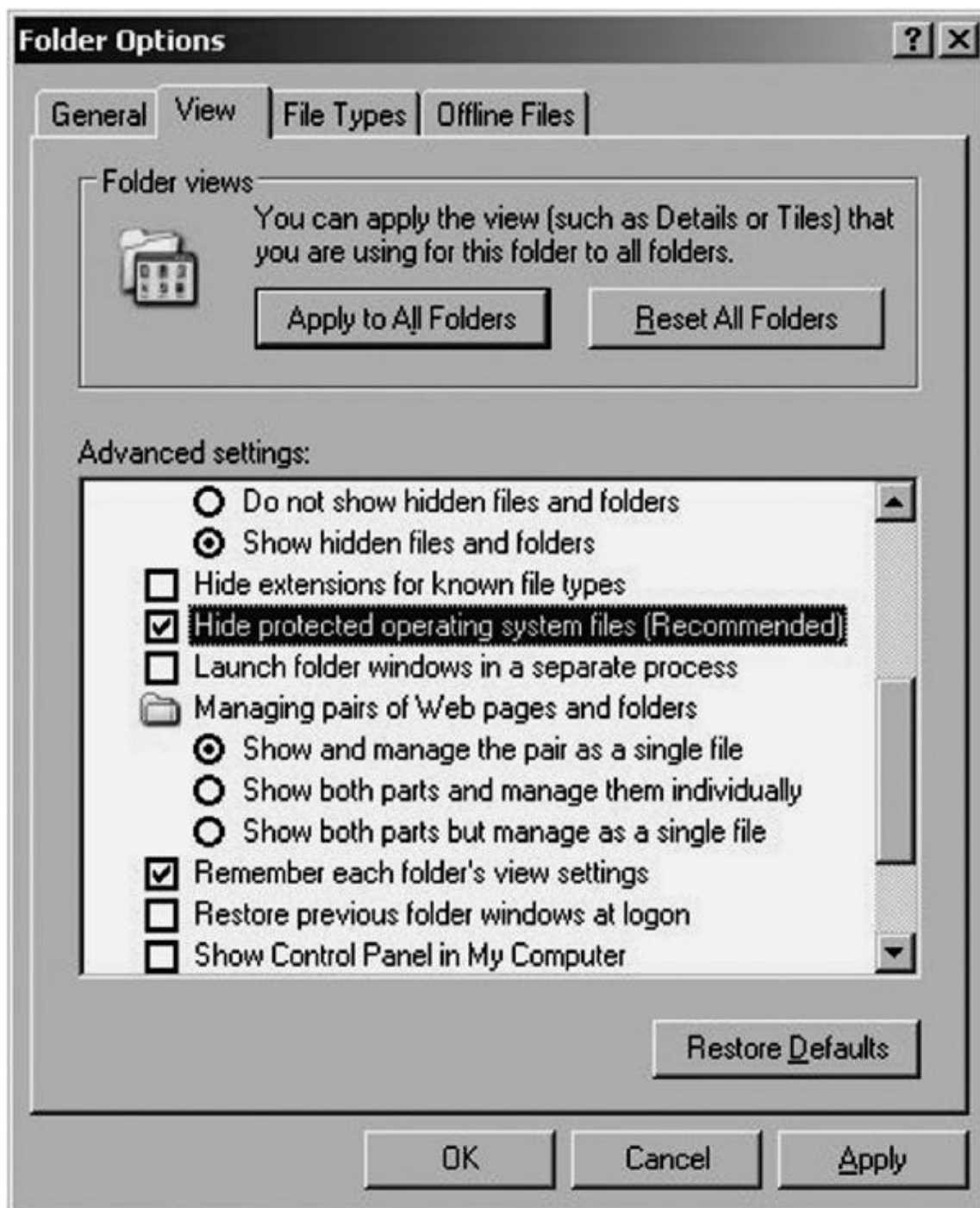
Thumbnails شکل دیگری از کش اطلاعات است که می‌توان بر روی رایانه مظنون بررسی کرد. Thumbnails در سیستم‌عامل ویندوز برای مشاهده‌ی سریع محتویات فایل‌های موجود در پوشه به کار می‌رود. بیشتر مردم نمی‌دانند که همزمان با مشاهده‌ی فایل‌ها در پوشه، فایل thumbs.db برای پیگیری تغییرات در همان پوشه ایجاد می‌شود. این فایل همچنین صفحه اول فایل‌هایی چون پاورپوینت را در خود ذخیره می‌کند.

اگرچه این اطلاعات، پوشه پنهان‌سازی داده‌ها را مشخص نمی‌کند، اما به بازرسان امکان پیگیری فایل‌های مظنون به پنهان شدن در فایل دیگر را می‌دهد. مثلاً اگر تروریستی نقشه‌ی ساختمانی را داشته

~~~~~

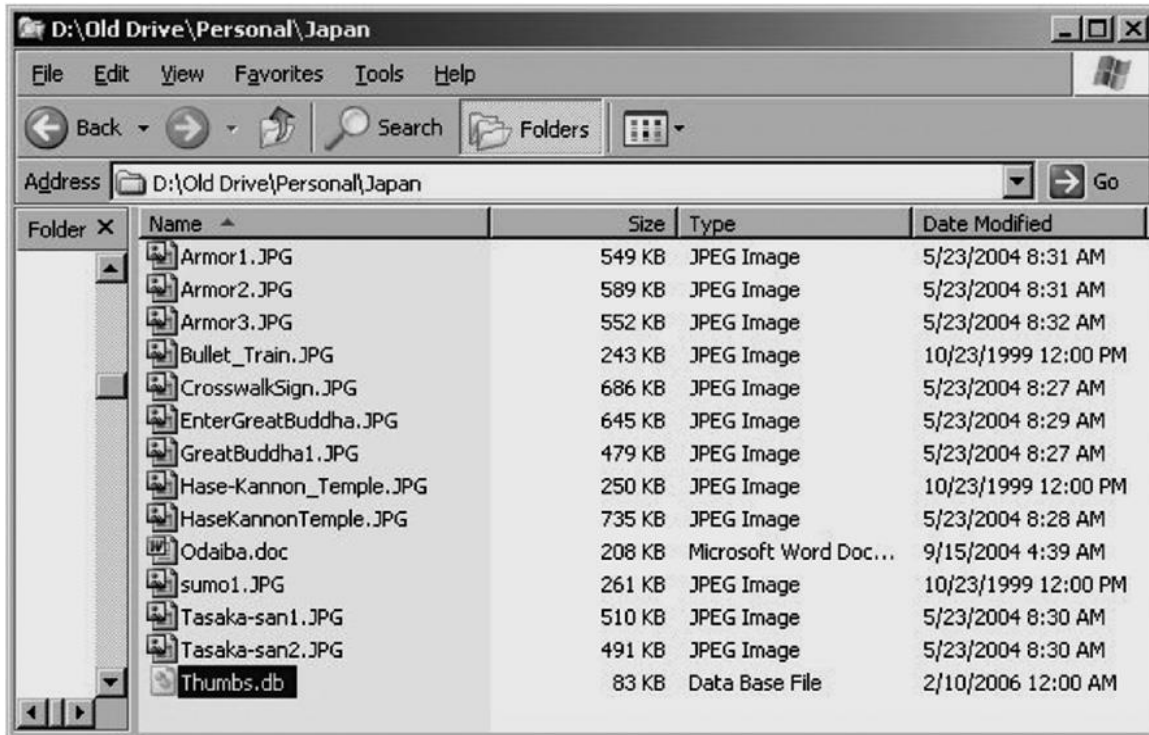
باشد و اقدام به پنهان کردن آن با استفاده از روش های استتار در فایل حاملی بنماید، نسخه ی کش شده از نقشه در فایل thumbs.db باقی می ماند. اگر تروریست به اندازه ی کافی عاقل باشد که برنامه ی استتار را حذف کند، باز به وسیله ی کپی نقشه ی باقیمانده در این فایل می توان رد نقشه و فایل حامل را گرفت.

برای مشاهده ی بایگانی داده های Thumbnails نخست در گزینه ی folder options گزینه ی view را انتخاب، سپس Hide protected operating system files را از حالت انتخاب خارج نمایید. (شکل ۱۰-۱۲) (در ویندوز ۷ در صفحه ی ویندوز اکسپلور، گزینه ی Organize را انتخاب سپس folder و پس از آن Search Options را انتخاب نمایید).



شکل ۱۰-۱۲: امکان مشاهده‌ی فایل **thumbs.db** به وسیله غیرفعال کردن این گزینه

مرورگر فایل در سیستم عامل ویندوز اقدام به نمایش فایل **thumbs.db** در پوشه‌ی جاری مثل شکل ۱۰-۱۳ می‌کند. شایان یادآوری است که این فایل همیشه در پوشه وجود دارد و فقط از دید کاربر پنهان است، مگر خود کاربر اقدام به نمایش آن کند.











شکل ۱۰-۱۳: فایل thumbs.db موجود در پوشه

نمایش عکس به شکل Thumbnails به صورت خودکار، یک کپی از عکس را در فایل thumbs.db موجود در همان پوشه ذخیره می‌کند، مگر این که کاربر به صراحت این گزینه را غیرفعال کند. این ویژگی در ویندوزهای XP، me و ۲۰۰۰ وجود دارد. تنها نکته در مورد ویندوز ۲۰۰۰ این است که در واقع این فایل در Alternate Data Stream ذخیره شده و در نتیجه در خود پوشه دیده نمی‌شود.

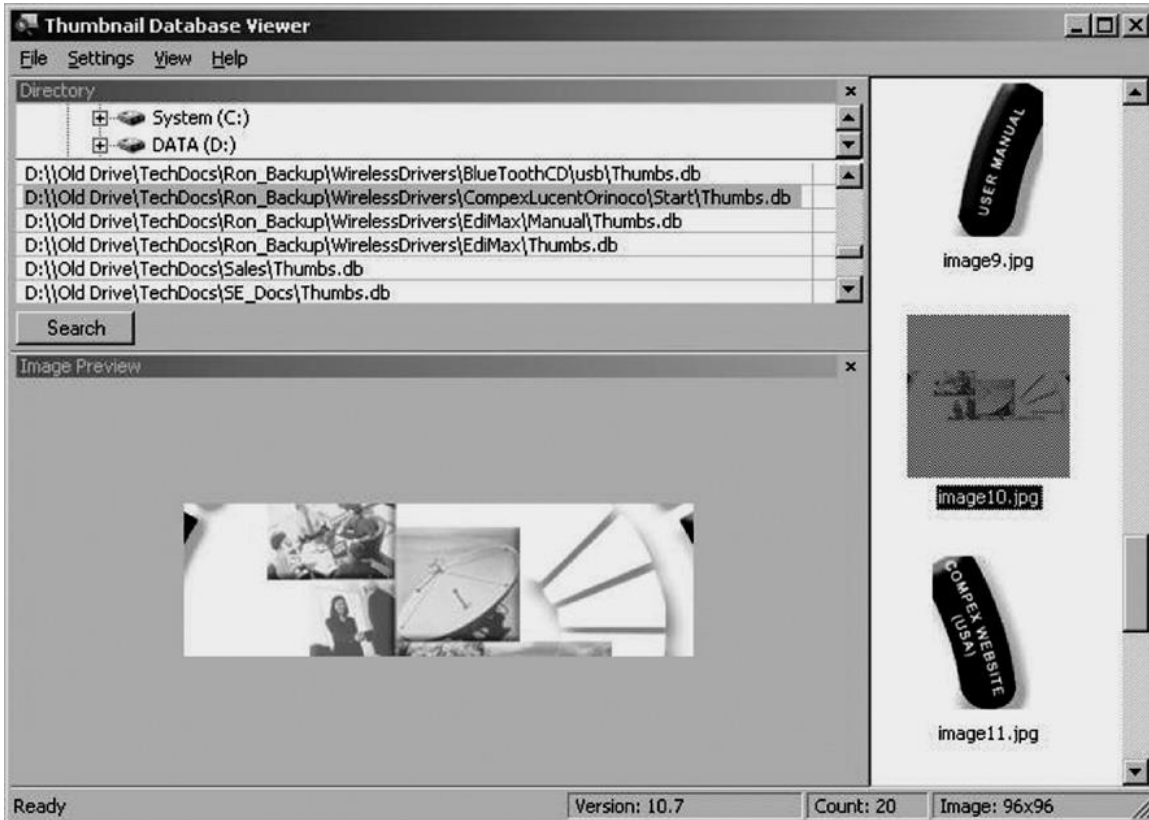
با روی کار آمدن ویندوز vista و ۷ فایل thumbs.db در محلی مرکزی، برای هر کاربر و در نشانی userprofile%\AppData\Local\Microsoft\Windows\Explorer\ (عکس ۱۰-۱۴).



| Name                                                                                                             | Date modified      | Type           | Size      |
|------------------------------------------------------------------------------------------------------------------|--------------------|----------------|-----------|
|  ExplorerStartupLog.etl         | 1/25/2011 1:39 PM  | ETL File       | 40 KB     |
|  ExplorerStartupLog_RunOnce.etl | 9/23/2011 9:13 AM  | ETL File       | 16 KB     |
|  thumbcache_32                  | 9/2/2011 1:06 PM   | Data Base File | 1,024 KB  |
|  thumbcache_96                  | 9/24/2011 11:07 PM | Data Base File | 10,240 KB |
|  thumbcache_256                 | 9/2/2011 1:06 PM   | Data Base File | 6,144 KB  |
|  thumbcache_1024                | 9/2/2011 1:06 PM   | Data Base File | 7,168 KB  |
|  thumbcache_idx                 | 9/19/2011 11:37 AM | Data Base File | 26 KB     |
|  thumbcache_sr                  | 9/2/2011 1:06 PM   | Data Base File | 1 KB      |

شکل ۱۰-۱۴: ویندوز ویستا فایل Thumbs.db را به صورت یک فایل مرکزی ذخیره می کند.

تعداد نرم افزار رایگان و تجاری برای مشاهده و تحلیل فایل های Thumbs.db وجود دارد. Thumbnail Database Viewer نرم افزار رایگانی برای مشاهده ی بانک اطلاعاتی فایل th و از نشانی <http://www.itsamples.com/thumbnail-database-viewer.html> قابل دانلود است. برای کار با این نرم افزار، فایل thumbs.db را که قصد مشاهده ی محتویات آن را دارید، انتخاب کرده یا از امکان جستجو برای لیست کردن تمام آن ها استفاده کنید. این نرم افزار امکان مشاهده ی محتویات فعلی و پیشین را داده و به حسابرسی و پیگیری تغییرات ایجاد شده در فایل های عکس، ویدئو، پاورپوینت و.... تا زمانی بر روی این سیستم انجام شده، می پردازد (عکس ۱۰-۱۵).



شکل ۱۰-۱۵: نرم‌افزار مشاهده‌ی محتویات پایگاه داده فایل

نکته‌ی جالب در خصوص thumbnails عکس‌هایی که در فایل thumbs.db ذخیره شده‌اند، این است که حتی پس از حذف فایل اصلی، باز هم در فایل thumbs.db باقی می‌ماند؛ به عبارت دیگر، برای همیشه در این فایل باقی می‌مانند، مگر اینکه به صورت دستی حذف شوند. اگر شخصی مظنون به اقدام وحشت‌افکنی باشد و تمام عکس‌ها موجود در سیستم را هم حذف کرده باشد، باز نسخه‌ی کوچک و مینیاتوری از آن‌ها هنوز در فایل thumbs.db وجود دارد.

اگر فایل سیستم را با استفاده از EFS رمزنگاری کنیم، سیستم‌عامل ویندوز با استفاده از thumbs.db نسخه‌ی مینیاتوری از عکس را به صورت رمزنگاری شده نمایش می‌دهد.

احتمال دارد که فرد مظنون، اقدام به انتقال عکس‌ها به حافظه‌ی قابل حمل نماید. این انتقال از حافظه‌ی اصلی به حافظه‌ی جانبی USB همچنان در پشت سرش فایل thumbs.db را در درایو اصلی باقی می‌گذارد؛ بنابراین مدارک حساس دال بر وجود فایل‌ها در رایانه مظنون باقی می‌ماند، حتی اگر حافظه‌ی قابل حمل هرگز پیدا نشود. این روش، مؤثرترین راه برای تشخیص عکس‌های موجود در سیستم است که زمانی در رایانه مظنون وجود داشته و می‌تواند بهترین راه برای crack هم باشد.

کاربران می‌توانند استفاده از thumbs.db را غیرفعال کند. مثلاً در ویندوز ۷ کاربر دستور رجستری خود را همانند شکل ۱۰-۱۶ در برنامه notpad اضافه کرده و سپس فایل را به نام disablethumbsdb.reg ذخیره می‌کند.

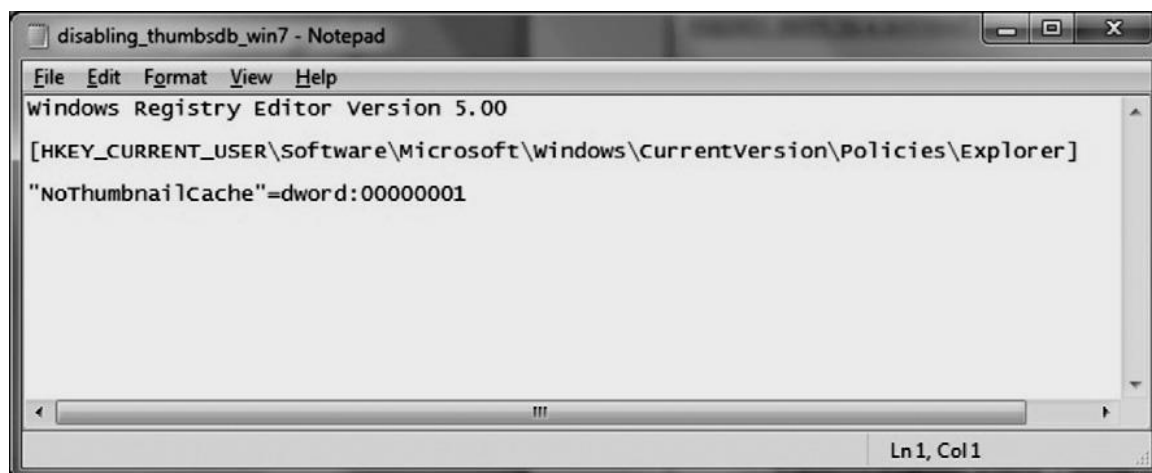
اکنون به سادگی و با دو بار کلیک بر روی نام فایل، محتویات آن به رجستری منتقل می‌شود، پس رایانه را reboot نمایید. «پیامدهای تغییر در رجستری را با مسئولیت خود انجام دهید».

## جستجوی فایل‌ها و پوشه‌های پنهان

در فصل پیش، در مورد Alternate Data Streams در ویندوز صحبت کردیم. ابزارهای مفید دیگری برای مشخص کردن فایل‌های پنهان شده در Alternate Data Streams وجود دارند. اجازه دهید نگاهی به نرم‌افزار LNS داشته باشیم.

## LNS

نرم‌افزار رایگان تحت ویندوز قابل دریافت از نشانی [www.ntsecurity.nu/toolbox/lns](http://www.ntsecurity.nu/toolbox/lns) برای یافتن فایل‌های پنهان شده در Alternate Data Streams بر روی رایانه میزبان بوده و جستجو را به صورت بازگشتی انجام می‌دهد. کار با آن به سادگی مشخص کردن نام درایو و پوشه مورد جستجو همانند شکل ۱۰-۱۷ می‌باشد.



شکل ۱۰-۱۶: استفاده از تنظیمات رجستری برای غیرفعال کردن Thumbs.db در ویندوز ۷

```

C:\WINNT\System32\cmd.exe
C:\Downloads\LNS>lsn c:\tools\ads

lsn 1.0 - (c) 2002, Arne Vidstrom <arne.vidstrom@ntsecurity.nu>
        - http://ntsecurity.nu/toolbox/lsn/

c:\tools\ads\mike.txt
        - Alternative data stream [:mikehidden.txt:$DATA]
c:\tools\ads\mike.txt
        - Alternative data stream [:mikehidden2.txt:$DATA]
C:\Downloads\LNS>

```

شکل ۱۰-۱۷: اسکریپت جریان‌های داده‌ای جایگزین Alternate Data Streams

به علاوه برنامه‌ی رایگان stream در نشانی از سایت مایکروسافت را می‌توانید دانلود نمایید.  
<http://technet.microsoft.com/en-us/sysinternals/bb897440>.

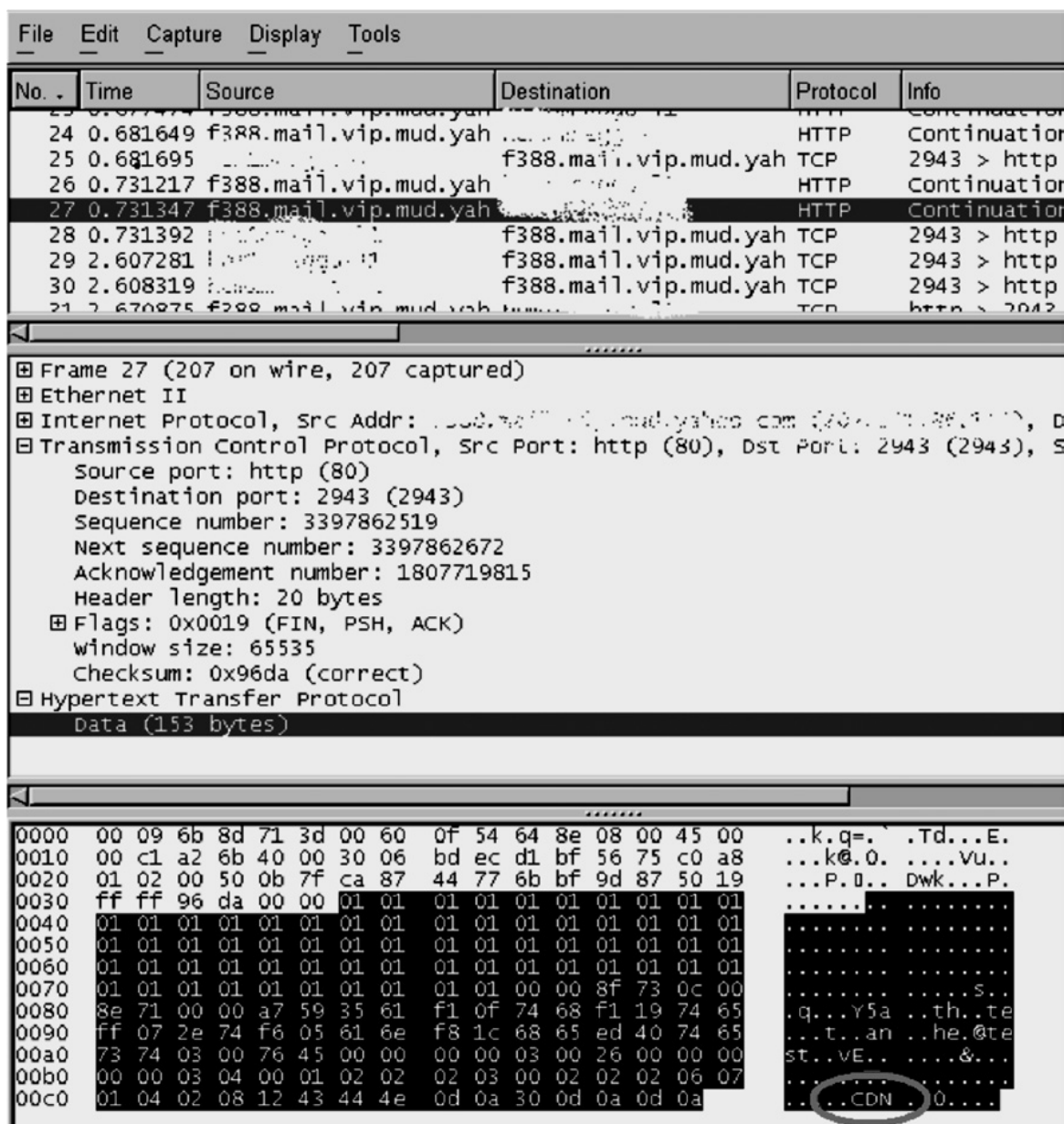
## سیستم تشخیص مهاجم در شبکه‌ها

با گسترش استفاده از سیستم تشخیص نفوذ در شبکه و سیستم‌های پیشگیری از ورود غیرمجاز در اکثر محیط‌های اداری و تجاری، شرکت‌ها نه تنها در فکر رویارویی با تهدیدات وارده به شبکه داخلی بلکه در فکر رودررویی با تهدیدات داخل سازمان مثل جاسوسی از شرکت، خرابکاری به بهانه‌ی اعتراضی و لو رفتن اطلاعات محرمانه هستند.

پایه و اساس تولید امضای دیجیتال برای شناسایی نرم‌افزارهای پنهان‌سازی داده‌ها شامل قوانین تولید شده و یک امضا یکتا به ازای هر برنامه استتار داده در نظر گرفته می‌شود. به علاوه این امضاها می‌توانند در نسخه‌های گوناگون نرم‌افزار هم متفاوت باشد؛ بنابراین نه تنها تشخیص امضا، بلکه خروجی گزارش به دست آمده که شامل برنامه‌ی شناسایی شده و نسخه‌ی برنامه می‌باشد نیز حائز اهمیت است. سایر اطلاعات مرتبط، مثل IP‌های منبع و مقصد IP‌ها را دربر می‌گیرد و بدین ترتیب در زمینه‌ی شناسایی ماشین مشکوک به بازرسان کمک می‌کند.

یک نرم‌افزار Shiffer می‌تواند برای به بوته آزمایش گذاشتن قوانین حفاظتی شما و تضمین حفاظت مناسب از شبکه بسیار مؤثر باشد. با استفاده از نرم‌افزار Wireshark می‌توانیم کانال ارتباطی را در حین انتقال یک فایل حامل یا انتقال مخفیانه شنود کنیم. شکل ۱۰-۱۸ کشفی را نشان می‌دهد که فایل حامل با امضای CDN را شناسایی کرده است. اگرچه احتمال دارد پایه این شناسایی فقط یک توالی

تصادفی از کاراکترها باشد، اما احتمال آن بسیار اندک است، بنابراین احتمال قوی‌تر این است که این امضای برنامه، «هایدرمان» باشد.



شکل ۱۰-۱۸: دستیابی Wireshark به یک امضای سرایندهمان

بنابراین می‌توانیم برای شناسایی این گونه فایل‌ها در آینده، یک امضای Snort بسازیم. یک نسخه از Snort از در سایت <http://www.backtrack-linux.org> با امکان نصب کامل در دسترس می‌باشد. وقتی Snort را نصب کردید، می‌توانید قواعد IDS داده‌های پنهان‌تان را به آن اضافه کنید. معمولاً قواعد IDS مقدار زیادی تشخیص اشتباه انجام می‌دهد؛ بنابراین حصول اطمینان از صحت قواعدتان حائز

اهمیت است (هرچه رشته‌ی امضا طولانی‌تر باشد، به همان نسبت دقت نیز بالاتر است). این قواعد می‌تواند در برگیرنده‌ی محتوای مطابق با نگاشت ASCII یا هگزادسیمال باشد. از آنجا که امضاها ی نرم-افزارهای پنهان‌کاری همیشه شامل نگاشت‌ها به ASCII نمی‌شود، معمولاً نمایش هگزادسیمال در این زمینه کارا تر است. دو راه تولید امضاء به هر صورت به شرح زیر است:

ترکیب امضا برای یک نماد ASCII:

```
Alert tcp any any <> any any (msg:"Message"; content:"content
```

ترکیب امضا برای نماد هگزادسیمال:

```
|Alert tcp any any <> any any (msg:"Message"; content:"| hex string
```

مثال زیر امضا برای «هایدرمان» در هر دو حالت ASCII و «هگزادسیمال» را نشان می‌دهد. لازم به ذکر است که امضا در بخش محتوای قانون قرار دارد.

```
Alert tcp any any <> any any (msg:"Hiderman Detected"; content:"CDN
```

```
Alert tcp any any <> any any (msg:"Hiderman Detected"; content:"43 44
```

```
4E
```

مثال قبل ممکن است به علت سادگی امضا، تعداد زیادی مثبت کاذب داشته باشد، اما مثال بعد یک رشته امضای طولانی‌تر برای برنامه‌ی پنهان‌سازی در JPEGX را نشان می‌دهد. امضا برای JPEGX ۷2.1.1 عبارت است از “۳۶ ۴۵ ۳۵ ۳۸ ۳۲ ۰۰ ۰۰”. نشان دادن این نوع امضا در قالب ASCII مشکل است، بنابراین تنها راه شناسایی آن، نمایش هگزادسیمال است. حاصل ایجاد یک امضاء برای JPEGX ۷2.1.1 به شرح زیر است:

```
Alert tcp any any <> any any (msg:"Jpegx ۷2.1.1 Detected"; content:"36
```

```
45 35 3B 00 00
```

اگر یک رشته‌ی نسبتاً پیچیده داشته باشید، نماد هگزادسیمال در مقایسه با رشته‌های ASCII در هر صورت می‌تواند دقت بهتری را فراهم کند. بنابراین، استفاده از هگزادسیمال بهترین راه است.

اگر بتوانیم به وسیله‌ی شنود از کانال، پنهان‌کاری را شناسایی کنیم، توانایی شناسایی بالقوه‌ی افراد مخرب بر روی شبکه را داریم. این کار با شناسایی فرد مظنون و کنترل فایل‌های مشکوکی که به وسیله اینترنت برای او فرستاده می‌شود امکان‌پذیر می‌باشد. فروشندگان محصولات تجاری سیستم تشخیص مهاجم، هنوز پایگاه امضاء برنامه‌های پنهان‌سازی را درست نکرده‌اند و اکثر فروشندگان سیستم‌های

تشخیص مهاجم (IDS) و سیستم‌های پیشگیری از نشت داده‌ها (DLP) برای شناسایی پنهان‌کاری بر روی شبکه، امضایی ندارند. شاید در سال‌های آتی این حوزه به رشد کافی برسد و به عنوان گزینه‌ی جدیدی برای شناسایی پنهان‌کاری پدیدار شود.

## چکیده

همان‌گونه که در این فصل اشاره شد، شخص مظنون به پنهان‌کاری، وظیفه‌ی بسیار چالش برانگیز پاک کردن کل سیستم از ردپاها را به عهده دارد. بازرس قضایی با استفاده از ابزاری که در اختیار دارد تلاش می‌کند برنامه‌های پنهان‌سازی داده‌های نصب شده را بیابد. به علاوه ممکن است فایل‌های کش ذخیره شده، فایل th.a و سایر مدارک و شواهد در رایانه هم پیدا شود. فرد مظنون اگر زرنج باشد، باید این فایل را حذف و فایل‌های اصلی را به حافظه‌ی جانبی قابل حمل منتقل کند شاید آن تنها راه منطقی دیگر نبود کردن ردپاها، خراب کردن فیزیکی رسانه‌ی ذخیره‌سازی یا Sanitization سطح پایین است. روش‌های بررسی شده در این فصل، بینشی در زمینه‌ی تکنیک‌های قانونی و غیرقانونی در هنگام رویارویی با پنهان‌سازی داده را در اختیار شما قرار می‌دهد.

~~~~~

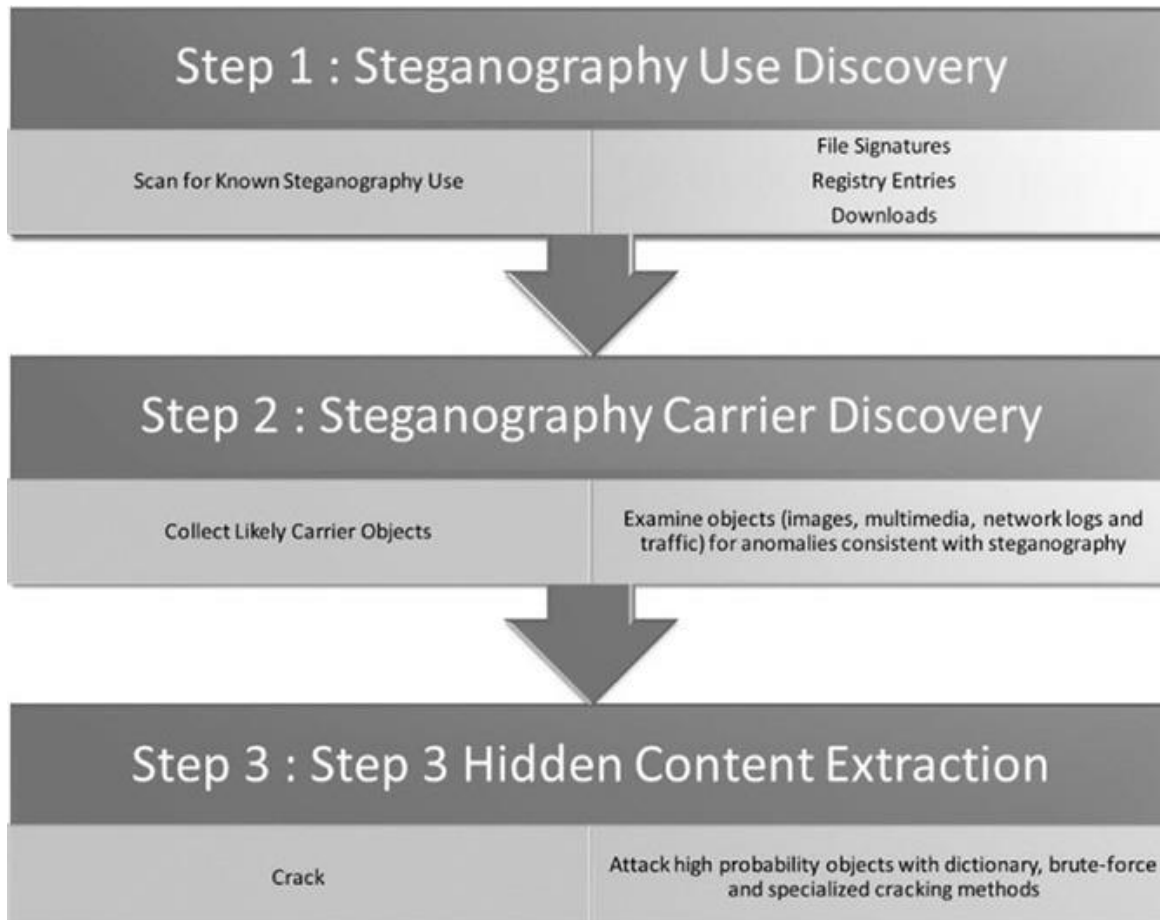


## فصل یازدهم

### بازرسی قضایی

همه ما ضربالمثل جستجوی سوزن در انبار کاه را شنیده‌ایم. از نظر تاریخی، این عبارت به سنت توماس مور در سال ۱۵۳۲ میلادی برمی‌گردد. وی نوشت «جستجوی یک خط در کتاب‌های او مانند جستجوی سوزن در یک مرغزار است. به همان سختی که به نظر می‌رسد.» چند سال پیش در کنفرانسی در دانشگاه جورج میسون، شناسایی پنهان‌کاری را به یافتن سوزن در انبار کاه تشبیه کردم. بلافاصله همان روز دکتر نیل جانسون که جزو حاضرین بود، گفته‌ی مرا اصلاح کرد. نیل گفت که توصیف دقیق‌تر «سعی برای یافتن قطعه‌ای نی در نیزار» است.

بدون شک در دهه‌ی گذشته با روش جمع‌آوری داده، آزمایش مستقیم آن‌ها و تجزیه و تحلیل ژرف‌تر برنامه‌های پوشیده‌نگاری شناخته شده در زمینه‌ی آشکارسازی داده‌های پنهان و فعالیت‌های آن‌ها در قالب استتار پیشرفت کرده‌ایم. اگرچه شیوه‌ی بررسی و شرایط اجرای آن گوناگون است. در نمودار زیر مدل اصلی شناسایی روش استتار و کشف پیام‌های پنهان را نشان داده‌ایم (شکل ۱۱-۱).



شکل ۱۱-۱: روش کلی و قانونی کشف استتار

### مرحله ۱: کشف کاربرد استتار

مرحله ۱، مستلزم دسترسی به محل ذخیره‌ی داده‌های مشکوک می‌باشد. باید یک کپی کامل (به وسیله نوشتن بلوک به بلوک) از ابزارهای ذخیره‌ی داده‌ی مشکوک ایجاد کنیم. این کپی‌برداری می‌تواند شامل ابزارهای ذخیره‌سازی محلی، ابزارهای ذخیره‌سازی از راه دور در شبکه، کارت‌های حافظه و غیره باشد. پس از کپی‌برداری، برای شناسایی برنامه‌های پنهان‌سازی داده یا استتار، پویش را آغاز می‌کنیم. در این مرحله نه تنها به دنبال فایل‌های اجرایی، بلکه در جستجوی فایل‌های هم زاد و موازی ورودی‌های رجستری مرتبط با برنامه‌های استتار شناخته شده هستیم.

در خلال این مرحله، تاریخچه‌ی کاوش در وب، برنامه‌های دانلود شده و جستجوهای انجام شده در اینترنت به وسیله‌ی مظنون که بیانگر علاقه وی به روش‌های استتار بوده را نیز بررسی می‌کنیم. این مرحله برای تسهیل پروسه در مراحل ۲ و ۳ از اهمیت بالایی برخوردار است. هرچه در مورد برنامه‌های استتار مورد استفاده مظنون بیشتر بدانیم، مراحل بعدی بازرسی هدفمندتر خواهد بود. به عنوان مثال،

اگر شواهد و مدارکی پیدا کنیم که مظنون برنامه‌های JP Hide و (JP HS) Seek را دانلود کرده است و مدارکی بر روی رایانه‌اش پیدا شود که سه روز پیش، این برنامه مورد استفاده قرار گرفته، مراحل بعدی بازرسی دقیق‌تر و حساس‌تر خواهد بود. از این گذشته، اگر این تنها برنامه‌ی استتار یافت شده ما باشد، می‌توانیم چنین نتیجه‌گیری کنیم که:

(۱) برنامه JP HS می‌تواند استتار را تنها بر فایل‌های JPEG انجام دهد (بنابراین دامنه‌ی جستجو را به فایل‌های حامل احتمالی محدود می‌کنیم).

(۲) آخرین تغییرات زمان دسترسی به فایل‌های JPEG در سه روز اخیر می‌تواند فایل‌های پوششی احتمالی باشد.

(۳) آخرین تغییرات زمان دستکاری فایل‌های JPEG در سه روز اخیر می‌تواند نشانگر جاسازی داده‌های پنهان تعبیه احتمالی باشد.

## مرحله ۲: کشف فایل حامل داده‌های پنهان

براساس نتایج مرحله ۱، فایل‌های حامل را جمع‌آوری می‌کنیم. این کار را می‌توان بر اساس نوع فایل حامل، زمان و تاریخ یا سایر داده‌های مرتبط فیلتر نمود. وقتی فایل‌های حامل احتمالی را جمع‌آوری کردیم، معمولاً در هر کدام از این فایل‌ها سه نوع تجزیه و تحلیل انجام می‌دهیم:

(۱) ابتدا الگوریتم‌های شناسایی ناهنجاری بر پایه‌ی امضا فایل را روی فایل‌های پوششی مظنون اجرا می‌کنیم. برنامه‌های استتار گوناگونی، ویژگی‌های فایل‌های حامل را به شیوه‌های قابل شناسایی تغییر می‌دهند. یک نمونه ساده، برنامه استتار Comouflage است که پس از نشانگر پایان فایل، داده‌ها را اضافه می‌کند. الگوریتم‌های شناسایی امضا، این ناهنجاری‌ها را به سادگی کشف و گزارش می‌کنند و فایل‌های متخلف را شناسایی می‌کنند.

(۲) سپس الگوریتم‌های شناسایی روش‌های استتار ناپیدا را که پیچیده‌تر است، اجرا می‌کنیم. این روش‌ها، مدل آمار فایل مشکوک را محاسبه و آن را با مدل‌های آماری شناخته شده از تصاویر و فایل‌های چندرسانه‌ای مقایسه می‌کند. هر نقطه کوری گزارش می‌شود و تجزیه و تحلیل‌های بیشتر پیشنهاد می‌گردد.

(۳) سرانجام تحلیل‌گر نتایج مراحل ۱ و ۲ را بررسی و سپس موضوعات را به صورت دستی مورد بررسی و آزمایش قرار می‌دهد. در این مرحله بررسی دیداری- شنیداری و بررسی چند شکلی روی فایل‌های مظنون انجام می‌شود. مثلاً اگر شک کنیم که پنهان‌سازی داده‌ها به شکل

جاسازی در مقادیر کم‌ارزش‌ترین بیت تصویر (BMP, True Color, PNG و غیره) انجام شده است، فقط مقادیر کم‌ارزش‌ترین بیت تصویر را پردازش می‌کنیم و به شکل دیداری مشخص می‌کنیم که آیا مقادیر موجود در کم‌ارزش‌ترین بیت داده‌های خود تصویر است یا این داده‌ها به روش‌های پوشیده‌نگاری جایگزین مقادیر واقعی شده‌اند.

### مرحله ۳: استخراج محتوای پنهان

وقتی برای رسیدن به درجه‌ی بالایی از قطعیت، دامنه را محدود کردیم، مواردی دیجیتالی داریم که حاوی اطلاعات پنهان هستند و می‌توانیم مرحله کرک را انجام دهیم. برعکس رمزنگاری که در آن برای فایل‌های رمزنگاری شده، استانداردهایی وجود دارد و این فایل‌ها باهم سازگارند، در استتار این‌گونه نیست. هر برنامه‌ی استتار با استفاده از روش‌های گوناگون، پنهان‌سازی را انجام می‌دهد. بنابراین می‌توان گفت که دانش زمینه‌ی برنامه‌های استتار، زیربنایی و پایه‌ای برای کرک محسوب می‌شود. رایج‌ترین شیوه‌های کشف برنامه استتاری که فرد مظنون از آن استفاده کرده است، به شرح زیر می‌باشد:

- (۱) در خلال مرحله ۱ (کشف کاربرد استتار) برنامه استتار کشف شده باشد.
- (۲) بر اساس ویژگی‌های به دست آمده در طول مرحله ۲ فرآیند (کشف فایل حامل استتار) برنامه استتار مشخص شده باشد.

وقتی برنامه‌های استتار احتمالی که ممکن است کاربر به‌کاربرده را تعیین کردیم و با گذر واژه شناخته شده (فرهنگ واژگان، یا به وسیله اطلاعات کشف شده از رایانه فرد مظنون) یا با استفاده از تولید گذرواژه به روش آزمودن کلیه حالت‌های ممکن فایل حامل را کرک کردیم؛ پس از انجام این مراحل، می‌توانیم با استفاده از برنامه‌ی شناسایی شده‌ی احتمالی یا ایجاد نرم‌افزاری که رفتار آن برنامه را در مد خودکار تقلید کند، هر گذرواژه‌ی احتمالی را آزمایش کنیم. سرانجام به علت ضعف ساختاری فایل‌های حامل یا نقص در روش‌های مدیریت کلید این فایل‌ها، می‌توان محتوای فایل‌های حامل مشخص را به طور مستقیم (بدون حدس گذرواژه) کرک کرد.

### کاهش پیامدها

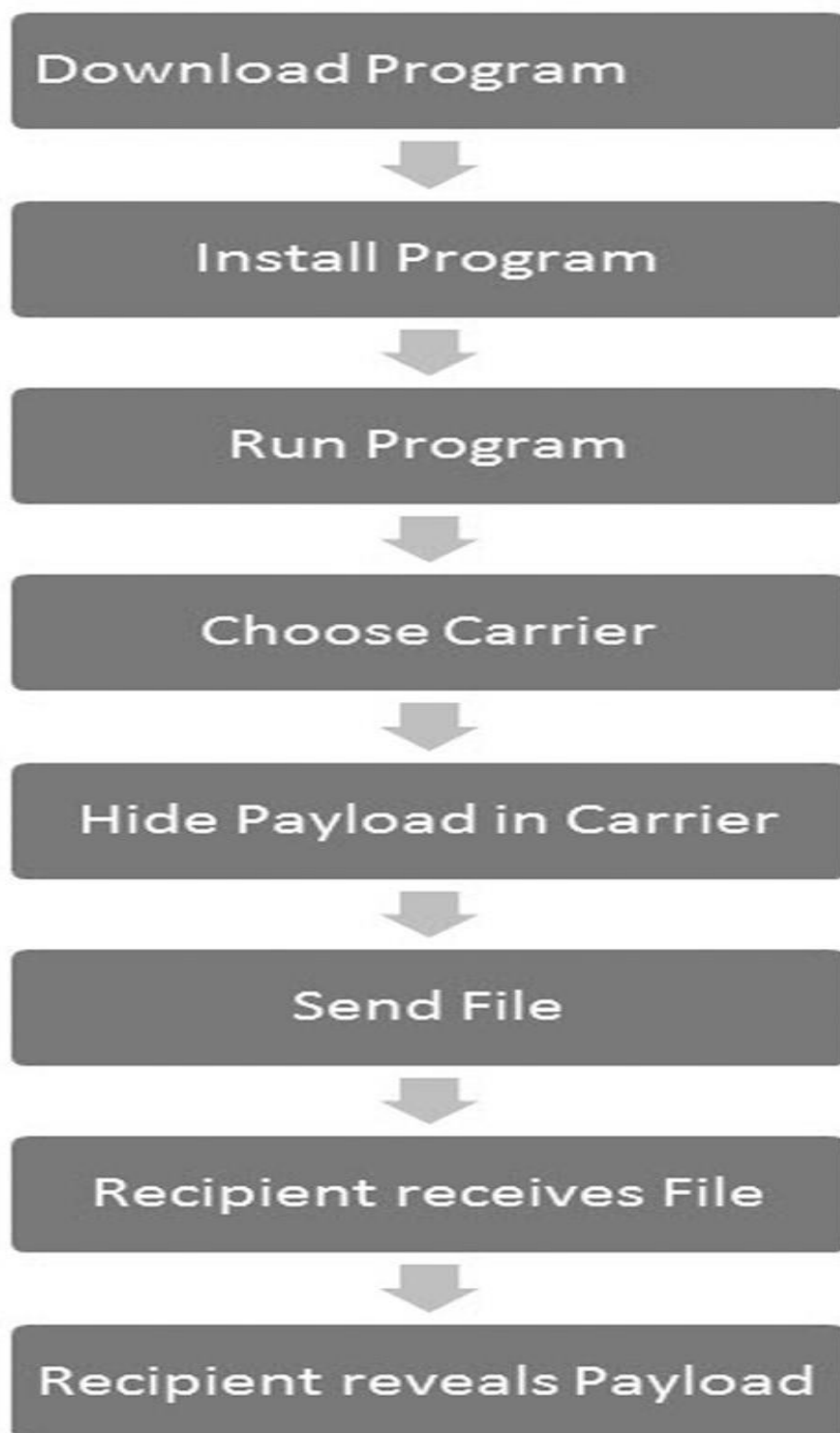
تعیین راه‌برد کاهش پیامدها، مستلزم درک کامل مواردی است که شما سعی در حفاظت در برابر آن دارید. در سال ۱۹۹۵، دن فارمر و ویست ونما، ابزار کاهش آسیب‌پذیری شبکه به نام SATAN (ابزار مدیریت امنیت برای تجزیه و تحلیل شبکه) را ارائه کردند. ایده‌ی فکری دن فارمر این بود که بهتر است

به جای تلاش برای هک کردن شبکه و استفاده از این اطلاعات برای ارتقای امنیت شبکه، از خود شبکه محافظت کرد. این تفکر، عصر جدیدی از امنیت شبکه با تمرکز به تست نفوذ و هک اخلاقی را به ارمغان آورد.

از این شیوه می‌توان برای پنهان کردن داده‌ها در شبکه برای تعیین میزان کارایی راهبرد دفاع در عمق استفاده کرد. با آزمایش برخی ابزارها و فن‌های بیان‌شده در این کتاب، می‌توانید میزان کارایی لایه‌های امنیتی به‌کاررفته در شبکه شما برای شناسایی و رویارویی با روش‌های گوناگون پنهان‌سازی داده‌ها را تعیین کنید.

فعالیت‌های پنهان‌سازی داده می‌تواند به جاسوسی از شرکت‌های بزرگ، ارتباطات پنهان، کارتابل استثمار کودکان، آسیب رساندن به داده‌ها و سایر فعالیت‌های مخرب منتهی شود. به عنوان مثال، شرکت‌ها نگران افشای اطلاعات شخصی کارمندان خود هستند و بیشتر از دستگاه‌های DLP (جلوگیری از نشت داده‌ها) استفاده می‌کنند و در مورد داده‌هایی که شبیه به داده‌های اطلاعات شخص کارمندانشان هستند سخت‌گیرتر می‌شوند و قوانینی وضع می‌کنند. اما وقتی با استفاده از روش‌های پیچیده، این داده‌ها پنهان شوند، شناسایی آن‌ها هم سخت‌تر می‌شود.

معمولاً انواع داده‌ها را چگونه پنهان می‌کنند؟ مثال‌های بیان شده در این کتاب، در مقایسه با گستردگی روش‌های به‌کاررفته در پنهان‌سازی داده‌ها، تنها نمونه‌ای از خروار بود. اما در اکثر موارد، شیوه و روش پنهان‌سازی داده‌ها از یک اصول کلی پیروی می‌کند. یکی از رایج‌ترین روش‌ها این است که کاربر، برنامه‌ی پنهان‌سازی داده را از اینترنت دانلود کند و از این برنامه برای پنهان کردن ظرفیت یا محتوای فایل حامل استفاده کند؛ سپس این فایل را برای استفاده کاربر مورد نظر و به وسیله‌ی پست الکترونیکی، ابزارهای به اشتراک‌گذاری و دانلود فایل ارسال نماید. اجازه دهید مرحله به مرحله نگاهی دقیق‌تر به این پروسه بیندازیم (شکل ۱۱-۲).



شکل ۱۱-۲: مراحل پنهان کردن داده با استفاده از نرم‌افزار

(<sup>۱</sup>) دانلود برنامه: رایج‌ترین شیوه‌ی پنهان‌سازی داده این است که کاربر نرم‌افزار پنهان‌سازی داده را دانلود می‌کند. این برنامه‌ها نه تنها فقط برای رایانه بلکه در نسخه‌ی موبایل آن نیز در دسترس هستند.

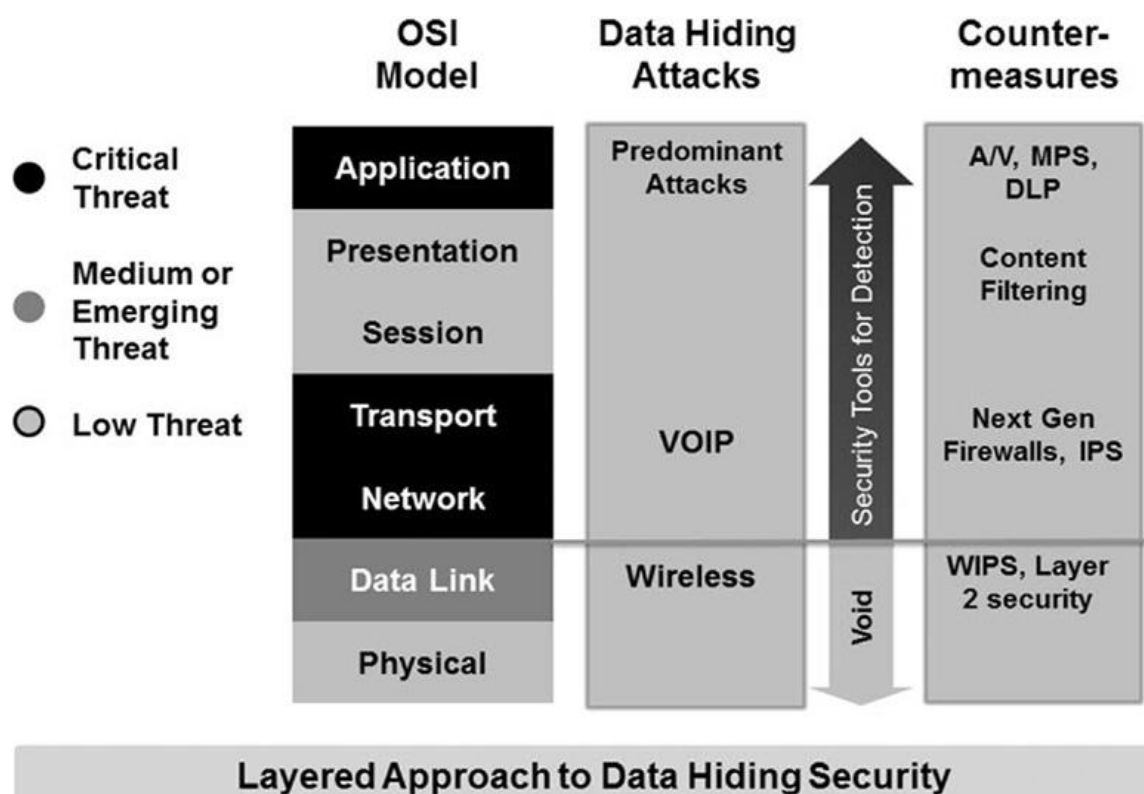
- ~~~~~
- (۲) نصب برنامه: زمان نصب برنامه، چند اتفاق در شبکه می افتد. نخست سیستم نصب برنامه می تواند شامل چند گام و با کپی کردن چند فایل همراه باشد. در سیستم عامل ویندوز ممکن است یک یا چند فایل dll نصب شود، ریجستری به روز شود و یک فایل اجرایی بارگذاری شود.
- (۳) اجرای برنامه: برخی برنامه ها پس از پنهان سازی داده ها، فایل اصلی را حذف می کنند.
- (۴) انتخاب فایل حامل: فایل حامل انتخاب شده برای پنهان سازی داده معمولاً به برنامه ی پنهان سازی داده بستگی دارد و از فایل JPEG تا فایل MP3 یا حتی فایل PDF می تواند متغیر باشد. در برخی موارد فرمت داده هایی را که می خواهیم پنهان کنیم را باید به فرمت فایل حامل تبدیل کنیم تا بتوانیم داده ها را پنهان کنیم؛ مثلاً، شاید لازم باشد فرمت یک نمودار را به گونه ای تغییر داد که برای برنامه ی حامل قابل فهم باشد؛ مثل تبدیل یک نمودار فایل Visio به فایل PDF.
- (۵) پنهان سازی داده ها در فایل حامل: به شدت به برنامه بستگی دارد. این مرحله شامل درج داده های پنهان و انتخاب گزینه های گوناگون دیگر ویژه ی آن برنامه می باشد. این گزینه ها می تواند شامل محل و چگونگی پنهان سازی داده، گزینه های پیش رو در رمزنگاری احتمالی و انتخاب گذرواژه باشد. وقتی که داده های پنهان درج شدند، فایل حامل ترکیبی از فایل اصلی و محتوای پنهان شده در آن می باشد، در نتیجه ممکن است در ترکیب فایل ها، ترتیب و حتی فرمت آن عوض شود (مثلاً تغییر فایل از شکل JPEG به BMP).
- (۶) ارسال فایل: اگرچه فایل حامل را می توان روی درایو قابل حمل منتقل کرد، اما معمولاً آن را به وسیله ی شبکه ارسال می کنند و می تواند به شکل ارسال به وسیله ی پست الکترونیک (ایمیل)، ارسال فایل به یک سایت برای دانلود آتی، آپلود آن به یک سرور FTP، یا جاسازی پنهان آن در یک پروتکل شبکه شود؛ بنابراین برای یافتن فایل حامل یا کشف رفتار غیرعادی آن، باید از چند ابزار شناسایی استفاده کرد.
- (۷) دریافت فایل: باز هم دریافت فایل مستلزم تعدادی گام دیگر در شبکه و در نتیجه احتمال بیشتری برای شناسایی فایل حامل است؛ خواه کاربر آن را از یک وب سایت دانلود کند یا به وسیله ی ایمیل دریافت نماید یا از یک سرور FTP، که باز هم فایل باید از شبکه بگذرد.
- (۸) آشکارسازی داده های پنهان دریافتی: به احتمال زیاد گیرنده نیز باید مانند فرستنده از همان برنامه برای بیرون کشیدن محتوای پنهان استفاده کند. این کار شامل تمام مراحل پیشین می باشد، در نتیجه پروسه ای دیگری برای امکان شناسایی گیرنده ایجاد می کند.

کاربر باتجربه ممکن است به جای استفاده از ابزار عمومی برای پنهان‌سازی داده، از تکنیک‌های پیچیده‌تر به شکل دستی استفاده کند. از آنجا که اجرای شیوه‌ی دستی پنهان‌سازی سخت‌تر است، این امر باعث شده است که شناسایی و کاهش پیامدهای پنهان‌سازی به این شیوه مشکل‌تر باشد، اما روش‌های به‌کاررفته بسیار شبیه به هم هستند. پنهان‌سازی داده به روش دستی فقط مراحل دانلود و نصب را در نمودار چرخه زندگی نرم‌افزار را ندارد ولی ممکن است چگونگی پنهان‌سازی داده در فایل حامل در روش‌های دستی و روش‌هایی که از این برنامه‌های نرم‌افزاری استفاده می‌کند، مشابه باشد. استفاده از روش‌های دستی فقط بر امکان شناسایی برنامه دانلود شده و تغییرات حاصل نصب نرم‌افزار چیره می‌شوند. در این کتاب بسیاری از تکنیک‌های دستی پنهان‌سازی داده‌ها را به تفصیل شرح دادیم. تاکنون به ذکر مراحل پنهان‌سازی داده پرداخته‌ایم. اما پرسش اصلی این است که چگونه باید آن را شناسایی کرد؟ شناسایی شامل ابزارها و تکنیک‌هایی است که داده در حال استراحت و داده در حال انتقال را تجزیه و تحلیل می‌کند.

### فناوری شناسایی داده‌های پنهان شده در شبکه

برخی شرکت‌ها از راه‌برد دفاع در عمق برای شناسایی و رویارویی با روش‌های پنهان‌سازی داده در شبکه استفاده می‌کنند. آمارها حاکی از آن است که با استفاده از ابزارهای رایگان موجود در اینترنت، بیشتر پنهان‌سازی داده‌ها انجام می‌شوند. بنابراین درحالی‌که انتقال داده‌های پنهان به شکل بی‌سیم یا استفاده از VOIP در قالب پوشش عملی است، اما برخورد با کاربر ویرانگری که از برنامه کاربردی شناخته شده پنهان‌سازی داده برای ارتکاب جرایمی نظیر استثمار کودکان یا جاسوسی از شرکت‌های بزرگ استفاده می‌کند، فراگیرتر است؛ بنابراین وقتی بین احتمال ارتکاب این جرایم و احتمال حمله VOIP یا بی‌سیم مقایسه‌ای شود، می‌توان نتیجه گرفت که پتانسیل آسیب از انجام چنین روش‌هایی بسیار بالاتر است (عکس ۱۱-۳).





شکل ۱۱-۳: معماری لایه‌بندی در امنیت برای رویارویی با پنهان‌سازی داده

امروزه محصولات گوناگونی برای شناسایی شواهد و مدارک و تکنیک‌های پنهان‌سازی داده وجود دارد. برای داده‌های در حال انتقال، ابزارهای مانیتور شبکه‌ای وجود دارد که تجزیه و تحلیل داده‌های گردآوری شده برای جستجوی داده‌های پنهان‌شده، مثلاً در فایل‌های PCAP را امکان‌پذیر می‌کند. علاوه بر این، محصولات نظارت جامع امکان مانیتور ارتباط در داده‌های چندگانه و ارائه آن را در یک صفحه SIEM (مدیریت رخداد اطلاعات امنیتی) فراهم می‌سازد. این محصولات بر پایه‌ی استفاده از ابزارهای تجزیه و تحلیل شبکه‌ای Live چون IPS و MPS (سیستم‌های حفاظت نرم‌افزارهای مخرب) استوارند. اما یافتن داده‌های پنهان، بسیار دشوارتر از شناسایی یک فایل مخرب شناخته شده است. از این گذشته، برای فایل‌های معمولی، نظیر عکس‌های معمولی، هش وجود دارد، اما وقتی شخصی داده‌ای را داخل عکس دیجیتالی ناشناخته‌ای پنهان کند، هشی وجود ندارد تا با مقایسه‌ی آن بتوان تعیین کرد که آیا عکس اصلی برای پنهان کردن داده دستکاری شده است یا نه.

جدول ۱۱-۱، انواع محصولات پرکاربرد در شبکه‌های سازمان‌های بزرگ را نشان می‌دهد که می-

توان از آن‌ها به عنوان بخشی از راهبرد دفاع در عمق برای شناسایی، کاهش و اصلاح فعالیت‌های پنهان-

سازی داده استفاده کرد که شامل داده‌های در حال انتقال در شبکه به شکل فایل و همچنین داده پنهان شده در خود و پروتکل‌های شبکه می‌شود.

جدول ۱۱-۱: فناوری‌های شبکه‌ای برای شناسایی فعالیت‌های پنهان‌سازی داده

| فناوری                                 | قابلیت‌های شناسایی، کاهش و زبان  |
|--|--|
| سیستم پیشگیری از نفوذ                  | شناسایی و بلوکه کردن دانلود برنامه‌های پنهان‌سازی و استتار داده (مسدود کردن برنامه)  |
| سیستم حفاظت در برابر نرم‌افزارهای مخرب | Sand box، تجزیه و تحلیل و مسدود کردن فایل‌ها و تجزیه و تحلیل ترافیک شبکه و بررسی قضایی و افزونه‌ها                                     |
| آنتی ویروس‌ها                          | قرنطینه و حذف برنامه‌های شناخته‌شده استتار و پنهان‌سازی داده‌ای.   |
| نسل بعدی دیواره آتش                    | امضاهای برنامه کاربردی<br>کد گشایی پروتکل برنامه کاربردی<br>به‌کارگیری روش‌های هوش مصنوعی در عملکرد شبکه                               |
| پیشگیری از نشت داده‌ها                 | شناسایی و مسدود کردن ارسال مدارک با ابرداده‌های خاص شرکت<br>شناسایی داده‌های پنهان‌شده در فایل‌ها (SS#، کارت اعتباری، PII، PHI و غیره) |
| سامانه پیشگیری از نفوذ بی‌سیم          | شناسایی دستکاری پروتکل بی‌سیم و مسدود کردن برون‌ریزی‌های مخرب  |
| فیلتر محتوا                            | شناسایی و مسدود کردن دانلود برنامه پنهان‌سازی داده   |
| متراکم سازی                            | پاک‌سازی و کدگذاری مجدد فایل‌ها  |
| مسدود کردن برنامه‌های کاربردی          | تهیه لیست سیاه <sup>۱</sup> و لیست سفید <sup>۲</sup> و تأیید هش برای برنامه‌های مطمئن شناخته شده                                       |

اکثر سیستم‌های پیشگیری از نفوذ<sup>۳</sup> (IPS) برای شناسایی روش‌های پنهان‌سازی و برنامه‌های استتار به خوبی تنظیم نشده‌اند. این سیستم‌ها برای شناسایی دانلود برنامه‌های استتار و پنهان‌سازی داده یا مرحله‌ی نخست چرخه‌ی زندگی نرم‌افزار مناسب هستند. در حال حاضر، بیشتر محصولات IPS فاقد لیست امضاها برای شناسایی اکثر برنامه‌ها پوشیده‌نگاری می‌باشند. بسیاری از برنامه‌های پرکاربرد پوشیده‌نگاری پروفایل شده‌اند و با کمترین هزینه می‌توان امضاء برنامه‌های جدید را به دست آورد و مانع دانلود آن‌ها شد.

۱ Blacklisting

۲ Whitelisting

۳ Intrusion Prevention Systems

سیستم‌های حفاظت در برابر نرم‌افزارهای مخرب برای تجزیه و تحلیل فایل‌های اجرایی ناشناخته با اهداف ویرانگر طراحی شده‌اند. با انتقال یک فایل اجرایی ناشناس به داخل sand box مجازی، MPS عملکرد فایل اجرایی را در زمان اجرا تجزیه و تحلیل می‌کند. هر فایل dll نصب‌شده یا دستکاری شده، تغییرات رجستری، سرویس‌های نصب‌شده و بسیاری از عملکردهای دیگر برای فعالیت‌های غیرمجاز و غیرعادی به معنی وجود بدافزار، در نتیجه شناسایی و تجزیه و تحلیل آن می‌شود. این تجزیه و تحلیل اکتشافی در عملکرد نرم‌افزار برای شناسایی برنامه‌های جدید استتار و پنهان‌سازی داده‌ای که پروفایل نشده‌اند، بسیار مناسب است.

چنین به نظر می‌رسد که آنتی‌ویروس‌ها، یک انتخاب قطعی و بدیهی برای شناسایی فعالیت‌های پنهان‌سازی داده و استتار باشد. گاهی برخی برنامه‌ها به جای ارسال فایل به وسیله‌ی ایمیل، آن را به وسیله‌ی اینترنت و از فایل سرور دانلود می‌کنند؛ بنابراین در این حالت، استفاده از آنتی‌ویروس برای خنثی کردن آن‌ها مناسب نیست، اما وقتی ایمیل‌هایی حاوی پیوست‌هایی با داده‌های پنهان شده‌ی ارسالی به وسیله‌ی برنامه‌های شناخته شده دریافت شود، استفاده از آنتی‌ویروس می‌تواند راه‌حل خوبی برای شناسایی و کاهش اثرات این‌گونه فعالیت‌ها باشد. امروزه کاستی اصلی آنتی‌ویروس این است که بیشترشان مجموعه جامعی از امضاها، برای شناسایی پیوست‌هایی با داده‌های پنهان شده ندارند.

نسل بعدی دیواره‌ی آتش<sup>۱</sup> با مجموعه‌ی از ویژگی‌هایی نوین، مکتب قدیمی دیواره آتش را تحت شعاع قرار داده‌اند. ویژگی کدگشایی پروتکل برنامه‌ی کاربردی این دیواره‌ی آتش به آن امکان شناسایی پروتکل جاسازی شده در داخل پروتکل دیگر و روش‌های گوناگون تونل‌زنی را می‌دهد. برای برخی از پروتکل‌های پنهان‌سازی داده، کد گشایی پروتکل برنامه‌ی کاربردی می‌تواند راه‌حلی احتمالی برای حل برخی از این روش‌ها باشد، اما در این حوزه باید پیشرفت‌های بیشتری صورت بگیرد. علاوه بر این، نسل بعدی دیواره آتش شامل تجزیه و تحلیل اکتشافی برای تجزیه و تحلیل روش‌های اکتشافی می‌شود. این روش‌ها دربرگیرنده‌ی پروتکل‌هایی است که از رمزنگاری اختصاصی و سایر روش‌های پوشیده‌نگاری استفاده می‌کند. اگرچه نسل بعدی دیواره آتش فاقد امکان کشف پنهان‌سازی و تشخیص انتقال به شکل پوشیده است اما می‌توان خط مشی‌های به‌کاررفته در این محصول را برای شناسایی برخی از این ارتباطات پوشیده پیکربندی نمود.

سیستم پیشگیری از نشت داده‌ها<sup>۱</sup> را می‌توان برای شناسایی فایل‌ها و اسنادی که ابرداده دارند مورد استفاده قرار داد. برای ردیابی اکستروژن‌هایی نظیر ابرداده‌های ناخواسته‌ای که نباید شبکه را ترک کند، می‌توان خط مش‌هایی ایجاد کرد. به طور کلی سیستم پیشگیری از ثبت داده‌ها برای شناسایی ابرداده‌های خاص شرکت یا مؤسسه نیازمند تنظیمات بیشتری می‌باشند. اما شایان به ذکر است که وقتی فایل یا مدارک دارای داده‌های پنهان شده از شبکه می‌گذرد، اکثر سیستم‌های پیشگیری از نشت داده‌ها توان شناسایی آن‌ها را ندارند. زیرا این فایل یا اسناد از برنامه‌های پنهان‌سازی داده با روش‌های پوشیده-نگاری استفاده می‌کنند. سیستم پیشگیری از نشت داده‌ها فقط امکان شناسایی تکنیک‌های ساده‌ای را دارد که داده‌ها در داخل فیلدهای ابرداده‌ای فایل‌هایی چون ورد، PDF و رایج و سایر برنامه‌های واژه‌نگار و صفحه گسترده پنهان شده‌اند.

## پارازیت غیر مخرب<sup>۲</sup>

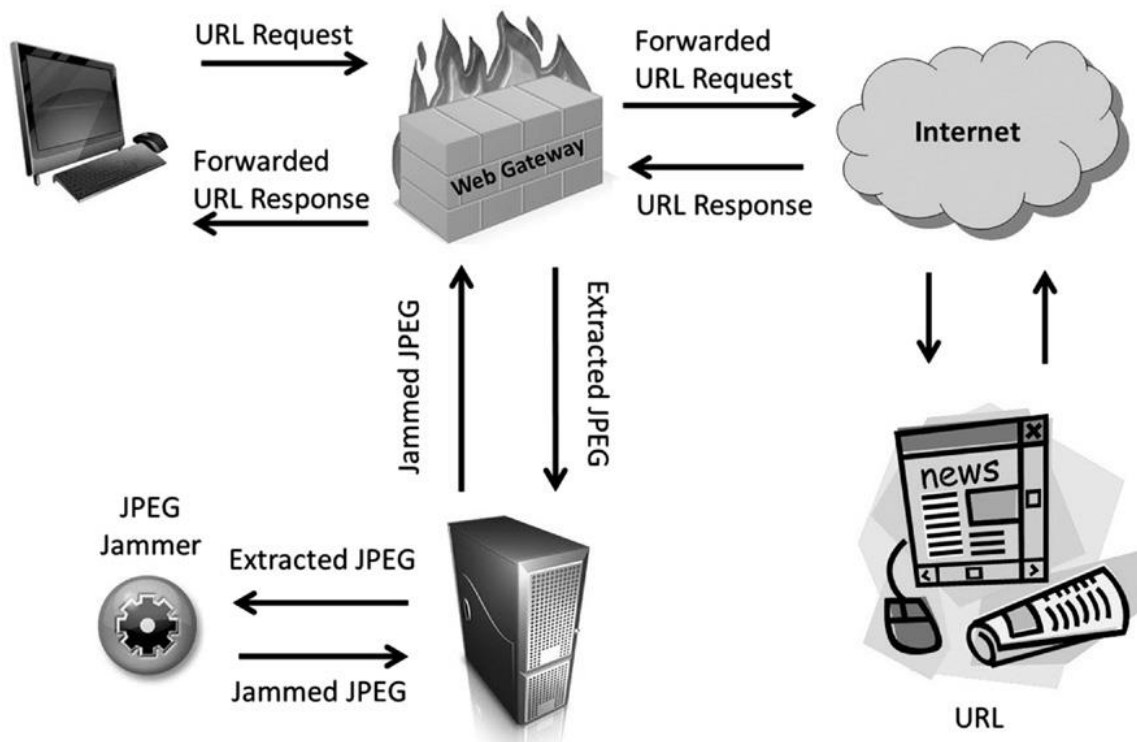
می‌توانیم ردپای یکی از اولین موارد به‌کارگیری در پارازیت‌ها در ارتباطات را در سال‌های ۱۹۰۴ تا ۱۹۰۵ میلادی در خلال درگیری بین روسیه و ژاپن پیدا کنیم. ایستگاه‌های تلگراف روسیه برای جلوگیری از ارتباط ناوهای جنگی ژاپن، پارازیت‌هایی را به طور تصادفی و مداوم بر کانال‌های ارتباط تلگرافی ژاپن ارسال می‌کردند. در طول جنگ جهانی دوم، بریتانیا و ایالات متحده بر پایه‌ی این پارازیت‌ها روش‌هایی را برای گریز از شناسایی به وسیله‌ی رادار گسترش دادند. نیروی هوایی برای گمراه کردن سیستم‌های رادار زمینی و هوایی آلمان، تکه‌های فلزی کوچکی را از هواپیما به پایین می‌انداختند. امروزه اقدامات متقابل برای رویارویی با ارسال پارازیت در نیروی دریایی و هوایی طراحی شده، زیرا گمراه کردن، بازی دادن و پنهان‌کاری در هر وضعیت میدان جنگ، حیاتی تلقی می‌شود.

استتار، برخی از مفاهیم اساسی پوشیده‌نگاری داخل زیرساخت‌های شبکه پیاده‌سازی شده است. در حملات Alureon Trojan, Shady Rat و بسیاری دیگر از برنامه‌های مخرب از روش‌های پنهان-سازی داده برای فرار از شناسایی در سیستم‌های پیشگیری از نشت داده، فیلترهای محتوا و برنامه‌ی دیواره‌ی آتش استفاده می‌کنند؛ بنابراین، نیاز به در نظر گرفتن شیوه‌های متراکم سازی و روش‌های توزیع ضروری است. هنگامی که تصاویر فایل‌های چندرسانه‌ای نقش پررنگ‌تری را در پیکان این‌گونه حمله‌ها به عهده گرفتند، سیستم‌های حفاظتی هم می‌توانند از روش‌های متراکم سازی غیر مخرب و کم‌هزینه

۱ Data Leakage Prevention

۲ Non-Destructive Jamming

برای حفظ کارایی شبکه استفاده کنند. نمونه‌ای از این روش، درگاه ورودی وب مجهز شده با JPEG Jamming است (شکل ۴-۱۱).



شکل ۴-۱۱: متراکم سازی غیر مخرب

پروسه بسیار ساده است:

- (۱) کاربر به وسیله‌ی مرورگر وب درخواست URL می‌دهد.
- (۲) مرورگر این درخواست را به یک URL در شبکه اینترنت ارسال می‌کند.
- (۳) URL به درخواست پاسخ می‌دهد.
- (۴) پاسخ به وسیله‌ی درگاه وب آزمایش می‌شود.
- (۵) اگر صفحه‌ی وب محتوا فایل JPEG باشد، پاسخ دریافتی از URL نگه داشته می‌شود و فایل JPEG به سرور JPEG Jamming ارسال می‌شود.
- (۶) سرور Jamming اقدام به متراکم و کدگذاری دوباره‌ی فایل JPEG می‌کند و هر محتوا پنهان کشف‌شده را هم گزارش می‌دهد.
- (۷) فایل JPEG تولیدی دوباره به درگاه وب ارسال می‌شود.
- (۸) سپس درگاه وب فایل JPEG را برای درخواست کننده می‌فرستد.

از دیدگاه کاربر فایل JPEG دریافتی تغییر نکرده است، ولی اگر درخواست صفحه وب از دیدگاه برنامه مخرب برای استخراج دستورهای جدید و اطلاعات کنترلی تعبیه شده در تصویر باشد، با تراکم فایل، این داده‌ها به شکل موفقیت آمیزی از بین رفته‌اند.

با این حال اشکال دیگری از متراکم سازی مورد نیاز است، زیرا اطلاعات پنهان را می‌توان به سادگی در فایل‌های دیگری مثل (فایل‌ها، تصاویر، فایل‌های چندرسانه‌ای، Web HTML، فایل وورد، صفحه‌های گسترده، جاوا اسکریپت و غیره) پنهان کرد.

### فناوری‌های نوین برای شناسایی پنهان‌سازی داده

بسیاری از تولیدکنندگان، برنامه‌ی کاربردی برپایه میزبان که قابلیت مسدود کنندگی دارد را ارائه می‌کنند. معمولاً این قابلیت‌ها مبتنی بر خط مشی تنظیمی است که از سوی سازمان یا شرکت تعیین می‌شود که کدام برنامه کاربردی مجاز<sup>۱</sup> و کدام برنامه کاربردی غیرمجاز<sup>۲</sup> است. این خط مشی را می‌توان بر اساس نام، امضا و هش یا حتی عملکرد برنامه‌ی کاربردی (مثلاً استفاده از سیستم تماس، حقوق مورد نیاز برای اجرای برنامه، مجوز کاربر و غیره) تعیین کرد.

بنابراین، این برنامه‌ها را می‌توان به سادگی برای جلوگیری از اجرای برنامه‌های پوشیده‌نگاری به کار برد که به وسیله‌ی بهبود خط مشی سازمان در اجازه ندادن به این‌گونه برنامه‌ها برای اجرا شدن در شرایط یا به وسیله‌ی کاربر خاص صورت می‌گیرد. تنها چیزی که لازم دارید فهرستی از امضاها (مقادیر هش) برای گردآیه‌ای از برنامه‌های استتاری است که قصد دارید اجرای آن‌ها را مسدود کنید (سپس این لیست را با برنامه‌های شناخته شده جدید به‌روز کرده) و از اجرای آن‌ها هم جلوگیری کنید. این سازوکارهای امنیتی چون سیستم پیشگیری از نفوذ میزبان<sup>۳</sup> Mcafee یا Symantic Critical Protection نه تنها سازوکارهای لازم مسدود کردن اجرای بدافزارها را فراهم می‌سازد، بلکه می‌تواند اقدام‌های انجام شده برای مدیریت کنسول‌ها یا سیستم‌های مدیریت اطلاعات امنیتی<sup>۴</sup> را نیز ارائه دهد و

---

۱ white listing

۲ black listing

۳ Host Intrastition Prevation System

۴ Security Information Event Management systems

کار پرسنل امنیتی را با دادن هشدار فوری و سطح بالا و نشت بالقوه داده‌ها به خاطر فعالیت بدافزارها را راحت کند (جدول ۱۱-۲).

جدول ۱۱-۲: فناوری‌های شناسایی فعالیت‌های پنهان‌سازی داده‌ها

|  |   |
|--|---|
| توانایی‌های شناسایی  | نقطه نهایی یا داده در آسایش               |
| سوءاستفاده برنامه‌ای<br>رفتارهای (عملکردهای) غیرعادی   | سیستم پیشگیری از نفوذ در میزبان<br>(HIPS) |
| تغییرات فایل‌ها و پوشه‌ها، تغییرات ریجستری، فایل‌های اجرایی<br>تغییرات در جدول‌ها و پیوست‌های پایگاه داده<br>تغییرات در محیط‌های مجازی<br>زمان REA یا برنامه‌ریزی شده  | نظارت انسجامی                             |
| برنامه‌های پنهان‌سازی داده نصب شده به صورت محلی  | آنتی ویروس                                |
| ابرداده‌ها و داده‌های الکترونیکی را محافظت می‌کند.   | کشف الکترونیکی                            |
| اسکن برای برنامه‌های استتار شناخته شده، dll‌های دستکاری شده ناسازگار<br>با آخرین dll شناخته شده  | پوشش آسیب‌پذیری (معتبر شده)               |
| شناسایی و مسدود کردن برنامه‌های استتار و پنهان‌سازی داده‌ها (مسدود کردن app)<br>مسدود کردن شبه DLP داده‌های بلوکه شده یا منتقل شده از یک doc به<br>فایل دیگر یا ارسال ایمیل به حساب شخصی کاربر   | امنیت و مدیریت دستگاه موبایل              |
| برنامه‌های پنهان‌سازی داده نصب شده به صورت محلی<br>مدارک و شواهد برنامه‌های پنهان‌سازی داده نصب شده در گذشته (dll‌های<br>حذف نشده، آثار به جای مانده ریجستری و غیره)<br>فایل‌های دارای محتوای جاسازی شده از برنامه‌های پنهان‌سازی شناخته<br>شده<br>فایل‌های دارای محتوای جاسازی شده از برنامه‌های پنهان‌سازی ناشناس<br>تجزیه و تحلیل ابرداده‌ای برای بررسی وجود داده‌های پنهان شده | نرم‌افزار forensics                       |

بسیاری از شرکت‌ها راه‌برد پوشش آسیب‌پذیری در شبکه‌های خود را اجرا کرده‌اند. برای اسکن‌های پوشش‌گرهای شبکه‌ای که امکان پوشش افزونه‌هایی مانند را هم می‌دهند، (مانند Nessus). مدیر سیستم می‌بایست کنترل‌هایی بر روی فایل‌های dll، فایل‌های اجرایی و سایر فایل‌های مربوطه به برنامه‌های پنهان‌سازی داده را هم در نظر بگیرد. شایان یادآوری است که معمولاً این موارد در خلال پوشش آسیب‌پذیری شبکه شناسایی نمی‌شود، بلکه در حین پوشش اعتبار سنجی کشف می‌شوند. امروزه بسیاری از

محصولات پوششگر آسیب‌پذیری از زبان اسکریپت نسویسی پشتیبانی می‌کنند. با دستورهای زیر می‌توان آزمایش پایه‌ای برای کنترل کلید رجستری ایجاد شده به وسیله‌ی برنامه‌های پوشیده‌نگاری مانند Camouflage، را فراهم ساخت.

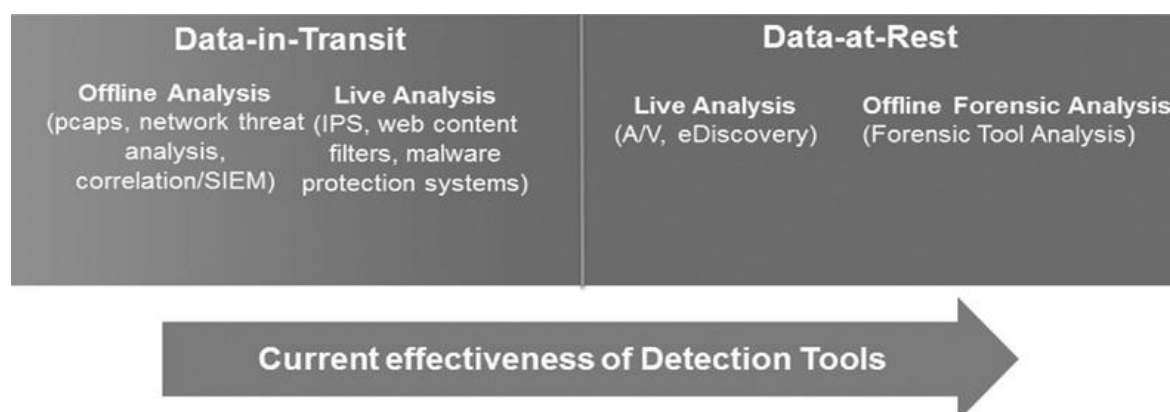
```
<if>
<condition type: "and">
<custom_item>
type : REGISTRY_SETTING
description : "steganography program Camouflage"
value_type : POLICY_TEXT
reg_key : "HKEY_CURRENT_USER\Software\Camouflage\CamouflageFile\0"
reg_option : CAN_BE_NULL
</custom_item>
```

همزمان با انقلاب تلفن‌های همراه، محصولات امنیت و مدیریت دستگاه موبایل به هدف شناسایی انواع روش‌های پنهان‌سازی داده و پوشیده‌نگاری، جایگاه بسیار خوبی پیدا کرده‌اند. بسیاری از این محصولات امنیتی، برنامه‌های استتار و پنهان‌سازی داده‌های موبایل را به سادگی شناسایی کرد و می‌توانند از دانلود این برنامه‌ها جلوگیری کنند. علاوه بر این اگر کاربر تلفنش را مجهز به یکی از محصولات مدیریت ابزارهای همراه<sup>۱</sup> کند، این محصولات بیشتر می‌توانند ارتباط او با شبکه شرکت را قطع کرده و بدین ترتیب کاربر را قرنطینه نمایند؛ از این گذشته، مدیر شبکه می‌تواند دستگاه را به طور انتخابی یا به صورت کامل حذف کند. همچنین محصولات مدیریت ابزارهای همراه شروع به گنجاندن ویژگی‌های شبه DLP در محصولات خود کرده‌اند تا به داده‌ای که نباید کپی شود امکان ارسال به وسیله‌ی پست الکترونیک یا فایل واژه‌پرداز متن را نداده و حتی امکان کپی برداری از پست الکترونیکی شرکت را هم ندهد.

## چکیده



امروزه بهتر است به جای شناسایی رفتارهای پنهان‌سازی داده در زمان انتقال، این داده‌ها را در مقصد و روی رایانه کاربر نهایی شناسایی کرد و این رویکرد، نتیجه تکامل ابزارهای شناسایی در طول زمان است. روش‌های شناسایی، آثار و شواهد به‌جامانده از وجود داده‌ی پنهان شده بر پایه‌ی تحقیق بازرسان قضایی می‌باشد. بنابراین، اکثر محصولات پیشرفته، ابزارهای تجزیه و تحلیل قانونی برای بررسی و آنالیز داده‌های موجود در رایانه‌های مشکوک را دارد (شکل ۱۱-۵).



شکل ۱۱-۵: کارایی ابزارهای شناسایی پنهان‌سازی داده

با گذشت زمان شرکت‌ها تمایل بیشتری برای محافظت در برابر حملاتی چون جاسوسی دارند. ابزاری نظیر آنتی ویروس‌ها، سیستم‌های پیشگیری از نفوذ بر پایه میزبان<sup>۱</sup>، کشف الکترونیکی، ابزار نظارت بر انسجام، همگی توانایی شناسایی این برنامه‌ها را بهبود می‌بخشد. برخی از فروشندگان آنتی ویروس شروع به اسکن محل‌های رایج پنهان‌سازی، نظیر جریان‌های داده‌ی متناوب کردند. ابزارهای کشف الکترونیکی پیشرفته امکان تعیین خط مشی‌های طبقه‌بندی فایل‌های معین و شناسایی وجود آن‌ها روی دسک‌تاپ‌های غیرمجاز را برای مدیران فراهم ساخته‌اند. اما محصولات تجزیه و تحلیل آنی داده موجود در رایانه<sup>۲</sup> بر اساس شناسایی شواهد و مدارک داده‌های پنهان شده یا عملکرد پنهان‌سازی داده‌ها، هنوز راه درازی تا تکامل دارد. مثلاً، سیستم‌های پیشگیری از نفوذ بر پایه میزبان و ابزار نظارت بر انسجام فایل، برای شناسایی سوء استفاده‌ی برنامه‌ای پیکربندی نشده‌اند. اگر کاربری فایل PDF را در Win Hex باز کند و داده‌ها را پنهان نماید، به احتمال زیاد این ابزارهای نظارتی توانایی بلوکه کردن این عملکرد غیرعادی را نخواهند داشت.

با در نظر گرفتن پهنه‌ی گسترده‌ی روش‌های پنهان‌سازی داده، به نظر می‌رسد که مدل کردن شیوه‌های پنهان‌سازی داده بسیار موثرتر از در نظر گرفتن تمام روش‌های پوشیده‌نگاری و چگونگی

<sup>۱</sup> Host-based intrusion prevention

<sup>۲</sup> Data-at-Rest Live Analysis

عملکرد آن‌ها خواهد بود. پس می‌توان سیستم‌عامل را ارتقاء داد و بدین ترتیب از رفتارها و عملکردهای ناخواسته‌ی بدافزارها جلوگیری کرد. به عنوان مثال، آیا کاربر عادی باید امکان پنهان‌سازی فایل‌ها در فایل سیستم جایگزینی را داشته باشد؟ آیا کاربر نهایی می‌تواند برنامه‌ی استتار شناخته شده را بر روی رایانه یا موبایل خود نصب کند؟ به وسیله‌ی سیستم‌عامل‌های موجود، خط مشی‌های مدیریت در دستگاه-های موبایل، امکان شناسایی، کنترل و ممنوع کردن این نوع رفتارها بسیار ساده است. این‌گونه رفتارهای کاربران به وسیله‌ی سیستم‌عامل یا پیکربندی سیاست مدیریت موبایل بسیار راحت‌تر قابل شناسایی، کنترل و منع هستند.

بسیاری از سناریوهای بیان شده در این کتاب را باید در یک شبکه‌ی آزمایشی پیاده کرد تا کارایی فناوری‌های دفاع در عمق در شبکه مورد بررسی قرار گیرد. به این آزمایش نخست، کارایی (یا ناکارایی) هر یک از محصولات مورد آزمایش قرار خواهد گرفت؛ سپس، تیم سازنده‌ی محصول با استفاده از نتیجه‌ی این آزمایش، محصول را تقویت، تنظیم و سفارشی می‌کند. روش وان فارمر را اجرا و با اعمال آن بر پنهان‌سازی داده، تیم طراح می‌تواند برنامه‌ی پنهان‌سازی داده‌ها را داندلود کرده و از آن برای پنهان کردن داده‌ها استفاده نماید و تعیین کند که آیا نسل بعدی دیواره‌ی آتش (NGFW)، فیلتر محتوای، آنتی ویروس، سیستم محافظت در برابر نرم‌افزارهای مخرب و سایر محصولات در شبکه‌ی این داندلود و به کارگیری آن را شناسایی می‌کند یا نه. در سایت [www.jitc.com/steganagraphy/tools](http://www.jitc.com/steganagraphy/tools) می‌توان لیست برنامه‌های رایج استتار را دریافت نمود. محصولات پیشرفته به مدیر امکان افزودن این برنامه‌ها برای شناسایی فیلترها را می‌دهد.

برای تعیین میزان کارایی سیستم پیشگیری از نفوذ مبتنی بر میزبان، نظارت انسجامی و سایر ابزارهای نظارت داده در استراحت، می‌توانیم برنامه‌ی استتار بی‌ضرر را نصب کرد و واکنش سیستم را ارزیابی کرد. هر آنچه را که شناسایی نمی‌شود را باید اصلاح و دوباره آزمایش کرد.

به طور خلاصه به نظر می‌رسد که فاصله‌ای در تجزیه و تحلیل رفتارهای کاربر مخرب روی لپ‌تاپ-ها و دسک تاپ‌ها وجود دارد. به عنوان مثال، آیا کاربر باید بتواند پنهان‌سازی داده در Volume Shadow Copy روی ویندوز لپ‌تاپ‌شان را انجام دهد یا یک پوشه‌ی لینوکس مخفی روی Mac آن‌ها ایجاد کند؟ یا کاربر باید فایل ورد را برای پنهان‌سازی داده در یک ویرایشگر مبنای ۱۶ ویرایش کند؟ شناسایی این‌گونه رفتارها باید در بیشتر محصولات امنیتی گنجانده شود تا کاربران مخرب از نادیده گرفتن فناوری‌های بومی بازدارد. برای حصول اطمینان از انسجام و اصالت، فایل‌ها باید امضا شده باشند.

این بررسی باید برای فایل میکروسافت ورد، فایل‌های Adobe PDF، چندرسانه‌ای‌ها و غیره انجام می‌شود. محصولات امروزی مجال بیشتری برای تکامل و ارتقای توانایی‌شان برای شناسایی روش‌های پنهان‌سازی داده دارند. این فرصت برای هر دو محصولی که روش‌های پنهان‌سازی داده را در داده در حال استراحت و همچنین داده در حال انتقال شناسایی می‌کنند، صدق می‌کند.

~~~~~

### نگاهی به گذشته و نیم‌نگاهی به آینده

استتار به واسطه‌ی ویژگی‌هایش، همواره در پهنه‌ی شیوه‌های پنهان‌سازی، حضوری فعال خواهد داشت. مقایسه‌ی استتار و رمزنگاری نشان می‌دهد که استتار دارای توانایی بیشتر در زمینه‌ی پنهان‌سازی و پنهانمانی تا زمانی است که دریافت کننده، آن‌ها را آشکار سازد. پنهان‌سازی داده‌ها کماکان به انتشار گسترده‌ی خود در تمام عرصه‌های زندگی روزمره‌ی ما ادامه می‌دهد. از RFID‌های پنهان در محصولات که می‌خریم گرفته تا چاپگرهایی که در صفحه‌های چاپ‌شده‌ی آن‌ها اطلاعات قابل شناسایی ولی پنهانی وجود دارد، همه و همه حاکی از نفوذ داده‌های پنهان در زندگی روزمره ما هستند. عمل پنهان‌سازی داده‌ها به شیوه‌های گوناگونی انجام می‌شود که حتی ممکن است از بسیاری از آن‌ها بی‌خبر باشیم.

هرچه از قرن ۲۱ می‌گذرد، شاهد رشد روزافزون دستگاه‌های موبایل و ارتباطات بی‌سیم هستیم و هر روزه چشم‌به‌راه پیدایش اشکال و کاربردهای نوینی از پنهان‌سازی داده‌ها هستیم. به دور از چشم کاربر عادی، عکس گرفته شده با تلفن هوشمند، دربرگیرنده‌ی مختصات GPS محل عکس‌برداری، نوع دوربین، شماره سریال تلفن و سایر اطلاعات قابل شناسایی پنهان شده در داخل عکس می‌باشد. اخیراً ارتباطات بی‌سیم دارای کد شناسایی در هدر بسته‌های شبکه‌ای شده‌اند. این کدها، جزئیات منبعی که بسته‌ها از آن ارسال شده‌اند را بیان می‌کنند. هر چند ما منتظر پیشرفت‌های بیشتری در ارتباطات موبایل هستیم. از دیدگاه کاربردهای نظامی، شاهد ارتباطات بی‌سیم قانونی و انتقال‌های مخفیانه در داخل تعداد زیادی از انتقال‌های ساختگی بوده‌ایم. به عنوان مثال، در سال ۱۹۴۲ میلادی بخش ضد اطلاعات رادیو<sup>۱</sup> کمسیون

---

<sup>۱</sup> Radio Intelligence Division (RID)

ارتباطات فدرال ایالات متحده<sup>۱</sup> برای شناسایی شبکه‌های جاسوسی آلمان، به نیروهای سرویس امنیت رادیو بریتانیا پیوستند. تجهیزات لازم شامل یک دستگاه گیرنده برای جمع‌آوری سیگنال‌ها روی دامنه‌ای از فرکانس‌ها و یک Snifter که مثل دستگاهی است که هنگام بازرسی از ساختمان‌ها برای شناسایی مبدأ سیگنال از آن استفاده می‌کنند و قابل حمل هم هست. این نیروها فرکانس‌هایی که طی چند ماه از انتقال‌های به ظاهر قانونی به دست آمده است را اسکن و بررسی کردند تا سیگنال‌های غیرعادی در اسکن‌های پیشین هم وجود نداشته باشد؛ سپس ناهنجاری‌ها کشف شده را برای بررسی مفصل‌تر به مراجع بالاتر گزارش دادند. با این کار، اطلاعات زیادی در مورد عملیات نازی‌ها جمع‌آوری شد؛ از جمله مدارکی دال بر وجود ابزاری با دستگاه فرستنده-گیرنده (به اندازه یک چمدان) و مجهز به آنتن‌های جهت‌دار، با قدرت سیگنال ضعیف برای کم کردن حوزه‌ی انتشار سیگنال که یک جاسوس نازی با آن، به همه جا سفر می‌کرد. با پیشرفت تجزیه و تحلیل‌ها، متفقیان انواع کدها را گردآوری و رمزگشایی کردند و با این رمزگشایی‌ها، محتوای بسیاری از پیام‌های نازی‌ها برای متفقیان آشکار شد.

وقتی نازی‌ها از تاکتیک‌های اکسیس (AXIS) آگاه شدند، آن‌ها نیز فونک سیپل<sup>۲</sup> خود را ایجاد کردند. فونک سیپل از کلمه‌ی فونک یا رادیو و سیپل یا بازی گرفته شده است. در اصل، نازی‌ها شبکه‌های رادیویی تله‌ای ایجاد کردند. این شبکه‌ها قانونی به نظر می‌رسید اما در واقع اطلاعات ساختگی ارسال می‌کردند. وقتی متفقیان سیگنال این شبکه‌ها را شناسایی کردند، باعث شد تا سربازان خود را آماده حمله نمایند. بدین ترتیب نازی‌ها، متفقیان را برای درک اهدافشان یا تعیین موقعیت حمله برای ضد حمله به بازی می‌گرفتند. این امر باعث شد متفقیان، اعتمادشان را نسبت به اطلاعات گردآوری شده از دست بدهند، تا اینکه آن‌ها نیز برای گمراه کردن نازی‌ها، بازی رادیویی خود را به راه انداختند. ممکن است امروزه هم بازی‌های رادیویی مدرن نیز گسترش یابد. این بازی‌ها می‌تواند به شکل انتقال‌های قانونی پنهان شده در داخل انتقال‌های ساختگی یا انتقال‌های گنجانده شده درون انتقال‌هایی که مانند در انداختن استراق سمع دور انداخته می‌شود.

## چالش‌های پیش‌رو

به گواه تاریخ، از دماراتوس در عصر باستان گرفته تا استفاده آلمان‌ها از رمزهای پوچ در جنگ جهانی دوم و جاسوسی روس‌ها در عملیات Shady Rat، همواره از پنهان‌سازی داده‌ها و شیوه‌های استتار استفاده شده و به مرور زمان هم تکامل یافته است.

<sup>۱</sup> United States Federal Communication Commission (FCC)

<sup>۲</sup> Funk Spiel

وقتی دستگاه‌های جدیدی مانند آی‌پد و اندروید عرضه می‌شود، تهدیدات پیشرفته‌ی پنهان‌سازی داده‌ها و استتار نیز به سرعت خودنمایی می‌کند و همان‌گونه که در زمینه‌ی تصاویر دیجیتال، فایل‌های چندرسانه‌ای، ماشین‌های مجازی و فایل‌های خود سیستم‌عامل مشاهده کردیم، پی‌درپی این فناوری‌ها دگرگون و به‌روز می‌شوند؛ بنابراین نیاز، تولیدکنندگان بدافزارها، رخنه‌گرها، سازمان‌های تبهکار، وحشت افکن‌ها و دولت‌ها برای پنهان‌سازی دستوراتشان و کنترل فعالیت‌های خود، قطعاً نه تنها شیوه‌های پوشیده‌نگاری، بلکه سایر روش‌های پیشرفته پنهان‌سازی داده‌ها هم گسترش و تکامل می‌یابند.

بنابراین چالش‌های پیش‌رو ممکن است شامل ترکیبی از روش‌های پیشرفته پنهان‌سازی داده‌ها در

حوزه‌های زیر باشد:

- ❖ محاسبات ابری
- ❖ مجازی سازی
- ❖ پروتکل‌های پیشرفته جریان داده‌ها
- ❖ ابرداده‌ها
- ❖ پایگاه‌های داده
- ❖ پروتکل‌های شبکه‌های بی‌سیم
- ❖ تبلت‌ها و تلفن‌های هوشمند

## فناوری بی‌سیم - یافته‌ی جدید

پروتکل‌ها فناوری‌های بی‌سیم با سرعت زیادی در حال رشد هستند به گونه‌ای که در مدت زمان کوتاهی بیش از پیش به نوآوری‌های روزانه در این حوزه عادت کرده‌ایم. وای‌فای، بلوتوث، 4G، G3 و گونه‌های دیگر این فناوری‌ها ما را در خصوص توانایی یا ناتوانی‌هایمان برای نظارت بر این فناوری‌ها در داده‌های پنهان شگفت‌زده می‌کند. فناوری‌های اندکی برای نظارت بر این شبکه‌ها در زمینه داده‌هایی که ممکن است دنباله بسته ارسالی (pay loud) یا حتی خود هدرهای پروتکل وجود دارد.

امکان انتقال داده‌های پنهان در پروتکل وای‌فای با استفاده از Beacons به عنوان راهکار انتقال داده وجود دارد؛ در واقع، شرکت مایکروسافت این روش را ایجاد کرد. نشان‌فرست Stuffing فنی است که در آن عناصر اطلاعاتی یک بسته نشان‌فرست وای‌فای را می‌توان برای انتقال تبلیغات به شکل توزیع شده در دستگاه‌های وای‌فای در حریم هوایی همسایه استفاده کرد. می‌توان آن را یک Kmart ویژه‌ی نور آبی تلقی کرد. کاربرد آن در فروشگاه‌ها به این شکل است که برنامه‌ی موبایلی را دانلود کنید تا از

قیمت‌های ویژه‌ی اجناس فروشگاه آگاه شوید. شبکه وای‌فای، آگهی‌های تبلیغاتی را در یک بسته‌ی نشان فرست و به جایی که موبایل کاربر وجود دارد برای وی ارسال می‌کند و کاربر با این برنامه می‌تواند میزان کوپن یا قیمت تخفیفی اجناس را برای کاربر مشاهده نماید.

با به‌کارگیری این شیوه در انتقال داده‌های پنهان، می‌توانیم از همین روش به طور کارآمد استفاده کنیم. بخش داده‌های این متن امکان استفاده از ۲۵۳ بایت از ۲۵۶ بایت را برای اطلاعات خاص فروشنده<sup>۱</sup> فراهم می‌سازد (شکل ۱۲-۱).

Insert Here

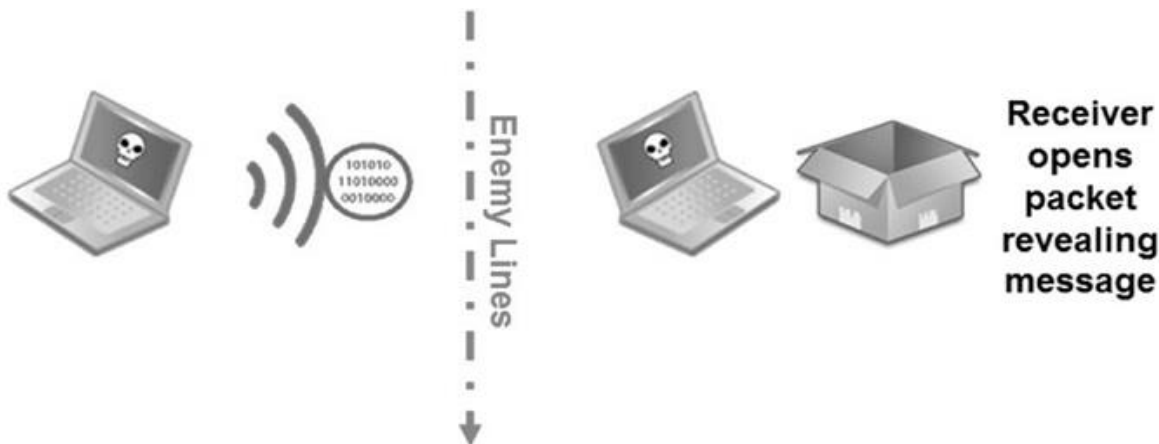


WiFi Beacon Packet

| Beacon Interval<br>(2 bytes) | Time Stamp<br>(8 bytes) | SSID<br>(32 bytes) | Supported Rates<br>(8 bytes) | Capability Info<br>(2 bytes) | Information Element<br>(256 bytes) | BSSID<br>(6 bytes) |
|------------------------------|-------------------------|--------------------|------------------------------|------------------------------|------------------------------------|--------------------|
|------------------------------|-------------------------|--------------------|------------------------------|------------------------------|------------------------------------|--------------------|

شکل ۱۲-۱: بسته نشان‌فرست وای فای

در این عصر مدرن می‌توان از دستگاه SPAC (ارتباطات عامل برد کوتاه) برای ارسال در یک سری از بسته‌هایی که خود حامل پیام بزرگ‌تر هستند، استفاده کرد، سپس پیام دریافتی روی دستگاه گیرنده دوباره به هم متصل می‌گردد. تا پیام ارسالی را بازآفرینی نماید (شکل ۱۲-۲).



شکل ۱۲-۲: Stego Stuffing

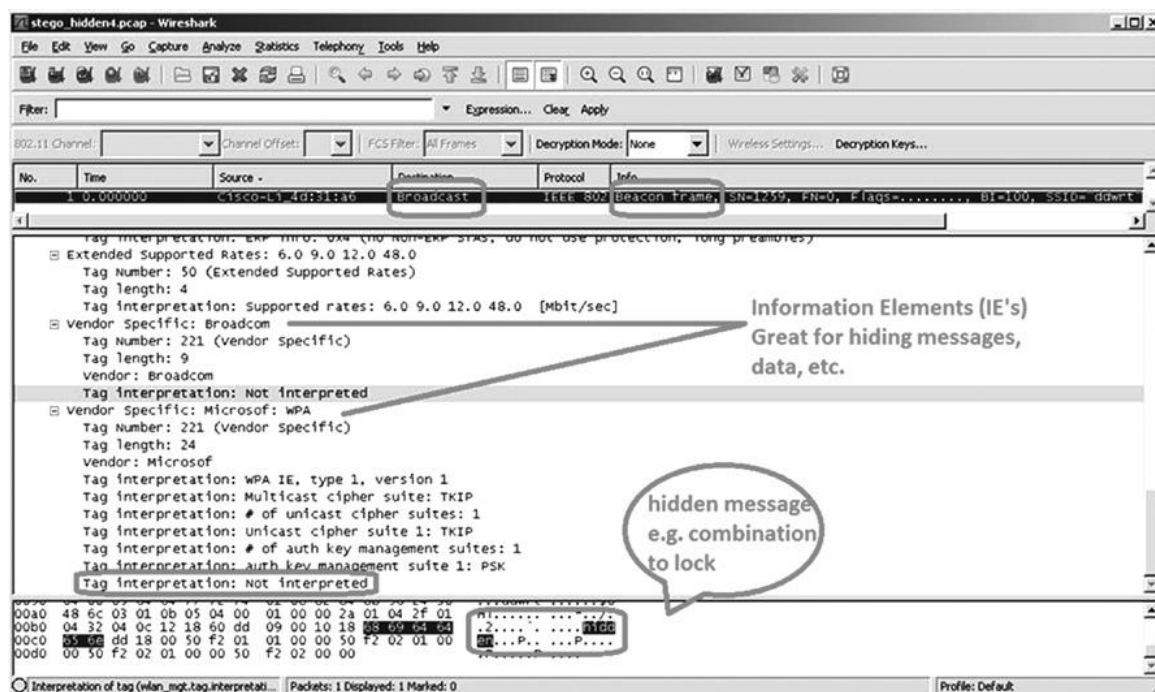
دستگاه SPAC را شوروی در سال ۱۹۷۰ میلادی اختراع کرد. ابتدا پیام روی رایانه نوشته می‌شد و سپس روی یک فرستنده کوچک SPAC منتقل می‌شد. این دستگاه که کمی بزرگ‌تر از یک بسته سیگار است، سیگنال معرفی برد کوتاهی می‌فرستاد. وقتی که گیرنده به اندازه‌ی کافی -حدود ۱۰۰ متر- نزدیک

<sup>۱</sup> Vendor- Specific information



می‌شد، فرستنده SPAC با آن ارتباط برقرار و پیام‌های منتظر ارسال را به شکل بخش همگانی را منتقل می‌کرد.

ترجیح می‌دهیم با از الهام گرفتن از مقاله‌ی Microsoft Research's Beacon Stuffing این روش را فن پیشرفته‌تر Stego Stuffing بنامیم. گیرنده می‌تواند با دریافت نتایج و بازبینی دوباره‌ی آن‌ها، پیام پنهان را استخراج و آشکار سازد. از این روش می‌توان برای فرستادن انواع پیام‌های کوچک، ترکیب‌های قفل‌گذاری شبه پیام‌رسان فوری و غیره استفاده کرد. پیام‌های بزرگ‌تر را می‌توان خلاصه و به یک‌باره ارسال نمود و یا با استفاده از ابزاری چون `aireplay-ng` آن را در چند مرحله فرستاد. روش‌های گوناگون ارسال به این شیوه بی‌نهایت‌اند. مثال زیر ارسال پیام با استفاده از Wire Shark را نشان می‌دهد (شکل ۱۲-۳).



شکل ۱۲-۳: بسته Stego Stuffing برداشته شده به وسیله‌ی نرم‌افزار Wireshark

علاوه بر پرس فرکانس در کانال‌های غیراستاندارد وای فای، سایر ویژگی‌های پروتکل بی‌سیم هم می‌تواند مانع شناسایی IPS بی‌سیم شود. به علاوه، در مثال فوق با دستگاه گیرنده‌ای که به هیچ شبکه‌ی بی‌سیم متصل نبوده و در برقراری ارتباط کاملاً به شکل بی‌قاعده انجام می‌داد که باعث مشکل‌تر شدن شناسایی فعالیت به وسیله‌ی شبکه بی‌سیم می‌شد، حتی می‌توان گفت که شناسایی گیرنده عملاً غیرممکن است.

بیشتر شرکت‌ها گردآیه‌ای امکانات را برای پیاده‌سازی راه‌برد دفاع در عمق برای شبکه‌های سیمی خود در اختیار دارند، اما وقتی به دستگاه‌های موبایل می‌رسیم که از نظر تعداد بیشتر از دستگاه‌های متصل به وسیله‌ی سیم است، باز هم با ضعف شناسایی داده‌های پنهان روبرو می‌شویم. این امر که بی-سیم، روش پایه‌ی برقراری ارتباط در میان خطوط دشمن است بسیار حائز اهمیت است. اگرچه سیستم‌های پیشگیری از نفوذ بی‌سیم بسیار کامل‌اند، اما این سیستم‌ها بیشتر برای شناسایی دستگاه‌های سرکش<sup>۱</sup>، حملات بی‌سیم به سایر دستگاه‌ها و سوء استفاده از پروتکل‌ها طراحی شده‌اند. در این حوزه، نیاز روزافزونی به فناوری‌های شناسایی بی‌سیم وجود دارد. حتی می‌توانیم منتظر پیدایش تکنیک‌های بیشتر پنهان‌سازی داده با استفاده از پروتکل‌های بی‌سیم شامل وای فای، بلوتوث، وای فای مجازی، G<sup>۳</sup>، 4G و غیره باشیم. همان‌گونه که در مورد اخیر جاسوسی روسیه، که از شبکه‌های موقت بی‌سیم برای انتقال اسناد سری استفاده شد، مشاهده می‌شود که این نیاز کماکان وجود دارد.

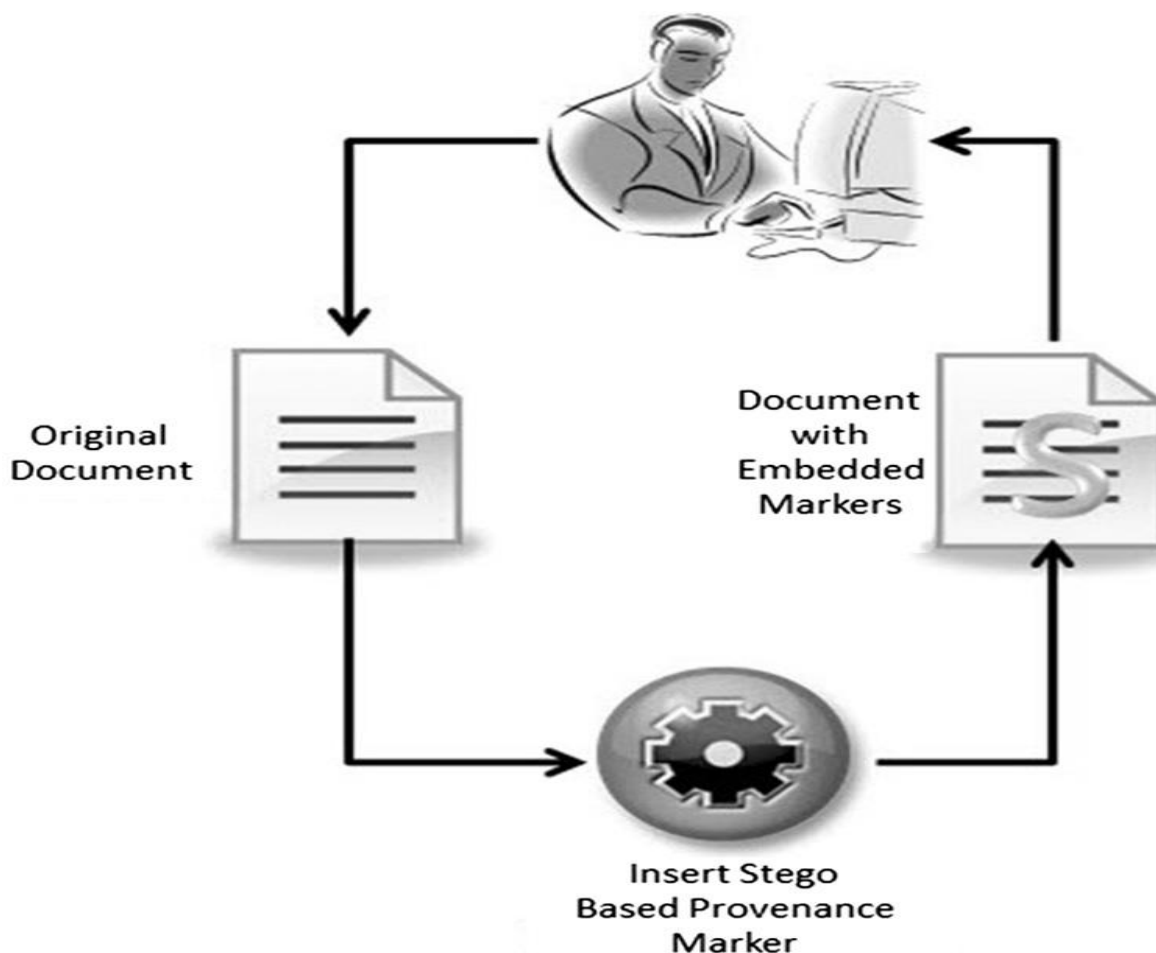
## استفاده از استتار در قالب یک اقدام متقابل

یک گزینه در زمینه‌ی دفاع از سیستم‌هایمان بازگشت به تاریخچه‌ای است که چگونه سیستم‌هایمان به وسیله‌ی خودی یا بیگانه مورد حمله قرار گرفتند. با به‌کارگیری توانایی‌های روش‌های استتار، به عنوان عمل متقابل می‌توانیم امنیت، ویژگی‌ها، ماهیت وجود و سرچشمه اسناد در شرکت‌های بزرگ، طرح‌ها، دارایی‌های با مالکیت معنوی و حتی پایگاه‌های داده‌ای که محتوا اطلاعات شخصی قابل شناسایی<sup>۲</sup> است را بهبود ببخشیم. شکل ۱۲-۴ نمای ساده‌شده‌ی این کار را نشان می‌دهد.

---

<sup>۱</sup> Rogue Devices

<sup>۲</sup> Personal Identifiable Information (PII)



شکل ۱۲-۴: استتار به عنوان یک عمل متقابل

کاربر مجازی می‌تواند یک سند، چکیده‌ای از اسناد و دستورالعمل‌ها، فایل صفحه گسترده، تصویر دیجیتالی، فایل چندرسانه‌ای و غیره را ایجاد می‌کند. فرض بر این است که این فایل‌ها باید حاوی نشانگر منشأ تولیدشان باشند. فایل اصلی به سرور Stego ارسال می‌شود و سرور نشانگرهای پنهان را در کل هدف می‌گنجاند. نشانگرها به گونه‌ای گنجانده شده‌اند که حتی اگر فایل دستکاری یا تغییر داده شود، باز نشانگرها باقی بمانند. این روش ممکن است شبیه واترمارک به نظر برسد، با این وجود، محتوای نشانگرها حاوی اطلاعات ماهیتی (مالکیت، محل، برچسب زمان، توضیحات، اطلاعات محرمانه، تاریخ انقضا و غیره) است و وقتی که سند، تصویر، فیلم، یا سایر موارد دیجیتالی در کل سازمان منتشر می‌شود، به شکل راهبردی بخش‌های امنیتی موجود در سازمان می‌توانند نشانگرها را شناسایی کنند و خط مشی اعمال نمایند که کنترل دسترسی، آشکارسازی، توزیع و اعمال انسجام فایل را تعیین کنند.

اسناد، تصاویر و غیره که فاقد نشانگرهای مبدأ هستند را می‌توان بر اساس قابلیت اطمینان و کنترل‌پذیری، پویش و نشانه‌گذاری نمود. حتی دستگاه‌های میزبان می‌توانند (بر مبنای خط مشی) تعیین

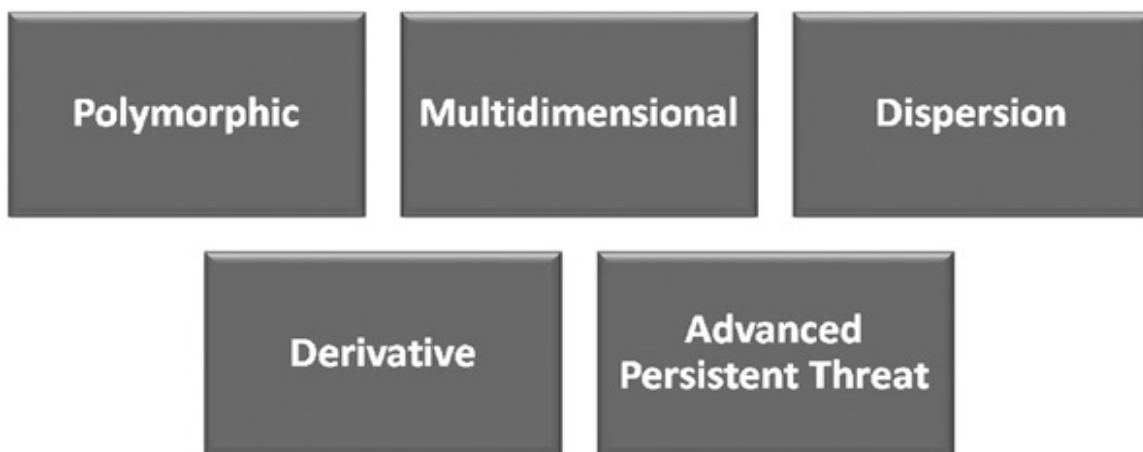
کند که چگونه موارد دیجیتالی یا بدون نشانگر منشأ را کنترل، قرنطینه یا پردازش نمایند، زیرا نشانگرهای پنهان تأثیری بر عملکرد عادی فایل ندارند (به عبارت دیگر، نشانگرها بر کیفیت تصاویر، فایل‌های چندرسانه‌ای، اسناد یا پایگاه داده‌ها تأثیری ندارند زیرا در کارکرد روزمره این اهداف مزاحم تلقی نمی‌شوند).

با آزمایش سودمندی شیوه‌های استتار برای چنین سطحی از محرمانگی، انسجام و برنامه‌های مورد اطمینان و قابلیت دسترسی در زیرساخت‌های سایبری خود را ارتقا می‌دهید.

بسیاری از سازوکارهای فعلی امنیت سایبری بر شناسایی غیرفعال تهدیدات استوار است. با افزایش تعداد شبکه‌ها و سرعت پردازش آن‌ها، اجرای این روش مشکل‌تر شده است و برخی از این سازمان‌ها، توزیع ترافیک در شبکه را گسترش داده‌اند. پس می‌بایست به وسیله‌ی داده‌های امنیتی و برچسب‌های الویتی که موجب بهبود کارایی شبکه خواهد شد، روش‌هایی هم برای کمک به پیاده‌سازی سازوکارهای امنیتی در شبکه فراهم کنیم.

## تهدیدات ترکیبی فعلی و آتی

انتظار می‌رود که پنهان‌سازی داده‌ها به روش‌های پیچیده‌تری به تکامل خود برای گریز از کشف و شناسایی ادامه می‌دهد. برخی از این روش‌ها، ترکیبی از دو یا بیش از دو مورد از این تکنیک‌ها می‌باشد (شکل ۱۲-۵).



شکل ۱۲-۵: تهدیدات مرکب

❖ چند شکلی: دقیقاً مانند ویروس، برنامه‌های پنهان‌سازی داده‌ها نیز می‌توانند خود دگرگونی انجام داده تا هنگام پنهان نمودن شناسایی شدن بر اساس امضای throw-off بگریزند (به

عنوان مثال Haydan که به شیوه‌های گوناگون داده‌ها را در فایل‌های اجرایی در زمان اجرا پنهان کند).

❖ چندبعدی: این نوع پنهان‌سازی داده از چند روش یا چند مرحله برای پنهان ساختن داده استفاده می‌کند. در چنین مواردی، نخست با استفاده از روش جاسازی داده‌ها در کم‌ارزش‌ترین بیت فایل حامل، داده را پنهان نموده، سپس این فایل را با استفاده از جریان داده‌ای جایگزین از دید سیستم‌عامل پنهان و یا در Volum Shadow Copy ذخیره نمود.

❖ پراکندگی: برخی برنامه‌ها برای پنهان کردن داده‌ها از چند فایل حامل یا چند ارسال گوناگون انتقال‌ها از روش‌هایی استفاده می‌کنند. از این گذشته، ممکن است در این شیوه، پنهان‌سازی انتقال‌ها یا فایل‌های تله‌ای بدون داده‌های پنهان فایل‌های ارسالی برای سردرگم کردن سیستم‌های حفاظتی اضافه می‌شود.

❖ مشتق: یک شکل از روش مشتق این است که داده‌ای در فایلی پنهان کرد، درعین حال سایر فایل‌ها را برای سردرگمی به شکل بررسی‌گر (throw-off) دستکاری نمود (به عنوان مثال فایل‌های زیادی را باز کرده تا آخرین تاریخ و زمان دسترسی همگی تغییر کند یا تعداد زیادی از فایل‌ها را تغییر داد.

❖ تهدید همیشگی و پیشرفته: اگرچه این تهدید تعریف گسترده‌ای دارد، اما نمونه‌ای جدید با عملیات Shady Rat که عکس‌های حاوی داده‌های پنهان را دریافت می‌کند و آخرین فرمان و کنترل URLها یا آدرس‌های IP در آن گنجانده شده است.

همان‌گونه که در فصل استراتژی‌های کاهش گفته شد، فناوری‌های شناسایی که الگوهای رفتاری و روش‌های اکتشافی مربوط به فعالیت‌های پنهان‌سازی داده را شناسایی می‌کنند، بهترین فناوری‌های سازگار برای رویارویی با این تهدیدهای ترکیبی محسوب می‌شوند. در یک جمله می‌توان گفت که راه رویارویی با تهدیدات مرکب استفاده از روش‌های شناسایی چندگانه است.

## چکیده

نگاهی به داستان دماراتوس و این که چگونه پیام سری را روی چوب حک کرد و برای گریز از شناسایی پیام، آن را به موم آغشته ساخت، از راه طولانی که به مدت ۲۵۰۰ سال طی شده است تعجب می‌کنیم. اخیراً برای گروهی از نخبگان امنیت سایبری که در زمینه‌ی حفاظت از دارایی‌های سازمان‌های خصوصی

و دولتی در سراسر دنیا فعالیت می‌کردند، کنفرانسی برگزار کردیم و پرسش‌های ساده زیر را مطرح نمودیم:

- اگر کسی یا برنامه‌ای اطلاعاتی را داخل یک عکس پنهان کند و سپس آن عکس را به عنوان پیوست به وسیله‌ی کانال‌های عادی با پیشرفته‌ترین محافظت‌های موجود امروزی ایمیل کند، چه مقدار از این پیام‌ها بلوکه می‌شود؟

در واقع این پرسش بیانگر تکرار ارسال پیامی به شکل پنهانی است که دماراتوس بیش از ۲۵۰۰ سال پیش با موفقیت انجام داد و تا به امروز هم ادامه دارد. حتی یک نفر هم برای پاسخ دادن اعلام آمادگی نکرد.

پایان