PORADNIKI

Wykrywanie treści steganograficznej w Internecie

1.Wprowadzenie

Steganografia jest sztuką i nauką ukrywania faktu ,że komunikacja ma miejsce. Systemy steganograficzne mogą ukrywać wiadomości wewnątrz obrazków lub innych cyfrowych obietktów. Dla przypadkowego obserwatora tych obrazków, wiadomość jest niewidoczna. W lutym 2000 roku, "USA Today" doniósł, że terroryści za pomocą steganografii ukrywali swoją komunikację. Według nich wiadomości były ukrywane w obrazkach wysyłanych na aulcje internetowe takie jak eBay czy Amazon. W artykule brak było jakichkolwiek inforamcji technicznych, które pozwalałyby sprawdzić te przypuszczenia. Aby ocenić twierdzenie ,ze treści steganograficzne są regularnie wysyłane do Internetu, potrzebujemy sposobu ,aby wykrywać steganograficzną zawartość obrazów automatycznie. Tu omówimy ramy wykrywania steganografii ,zaczynając od robota internetowego ,który zbiera obrazki JPEG z Internetu. Skorzystamy z analizy statystycznej, podzbioru obrazów, które mogłyby zawierać treści steganograficzne. Analiza jest statystyczna, tzn . nie gwarantuje ,że określone obrazy zawierają naprawdę ukrytą treść, więć opiszemy również środowisko obliczeń rozporoszonych, które uruchamia atak słownikowy umieszczony w klastrze lużno połączonych stacji roboczych w celu wykrycia ukrytej zawartości. Omówimy analizę dwóch milionów obrazków pobranych z aukcji eBay. Do tej pory nie udało się znaleźć ani jednej wiadomości. W części 2 omówimy krótko steganografię. Część 3 wyjaśnia jak ukrywać informacje w obrazkach JPEG. Część 4 przedstawia testy statystyczne umożliwiające wykrywanie zawartości steganograficznej. W części 5 podamy omówienie istniejących sysytemów steganograficznych i opiszemy jak je wykrywać. Wykrywanie kontekstu jest omówione w części 6. Wyniki i związaną z nimi pracę omówimyu w części 7.

2.Trochę o steganografii

Termin "Ukrywanie Informacji" odnosi się zarówno do znaku wodnego jak i steganografii. Zank wodny zwykle odnosi się do metod ,które ukrywają informację w danych obiektu, więc informacje te są odporne na modyfikację. Oznacza to ,że powinny być niemożliwe usunięcie znkau wodnego bez degradacji jakości danych obiektu. Z drugiej strony, steganografia odnosi się do ukrywania informacji, które są wrażliwe. Modyfikacja częściowa może je zniszczyć. Zank wodny i steganografia różnia się w inny istotny sposób: podczas gdy informacje steganograficzne nie muszą być oczywiste dla oglądającego, ta funkcja jest opcjonalna dla znaku wodnego. Bezpieczeństwo klasycznego systemu steganograficznego opiera się na tajemnicy systemu kodowani. Kiedy system kodowania jest znany, system zostanie pokonany. Znanym przykładem klasycznego systemu jest ten, w którym Rzymianie golili głowę niewolnika i tatuowali ukrytą wiadomość, a kiedy włosy odrosły, niewolnik był wysyłany z wiadomością. Taki system może działać, do momentu kiedy nie zostanie rozpoznany, wtedy goli się głowy wszystkich ludzi aby sprawdzić ukryte wiadomości. Inny system kodowania może wykorzystywać ostatnie słowo w każdym zdaniu lub najmniej znaczące bity w obrazie. Jednak nowoczesna steganografia powinna być wykrywalna tylko, jeśli znana jest tajna informacja, a mianowicie tajny klucz. Jest to podobne do "Zasad Kerckhoffa" w krytpografii, które stwierdzają ,że bezpieczeństwo systemów kryptograficznych powinno zależeć od kluczowego materiału. Ze względu na inwazyjną naturę, systemy steganograficzne pozostają wykrywalnymi śladami wewnątrz właściwości medium. Pozwala to na wykrycie podsłuchu modyfikacji mediów, ujawniając tajemnice komunikacji. Chociaż tajna zawartość nie jest eksponowana,jej istnienie zostaje ujawnione, co rujnuje główny cel steganografi. Generalnie, proces ukrywania informacji składa się z następujących kroków:

1. Identyfikacji nadmiarowych bitów w danym medium. Nadmiarowe bity są tymi bitami, które mogą być modyfikowane bez utraty jakości
2. Wyboru podzbioru nadmiarowych bitów zastępowanych danymi tajnej wiadomości. Medimu steganpograficzne jest tworzone przez zastąpienie wybranych bitów nadmiarowych bitami wiadomości

Modyfikacja nadmiarowych bitów może zmienić statystyczne właściwości medium. W wyniu, analiza statystyczna może ujawnić ukrytą zawartość.

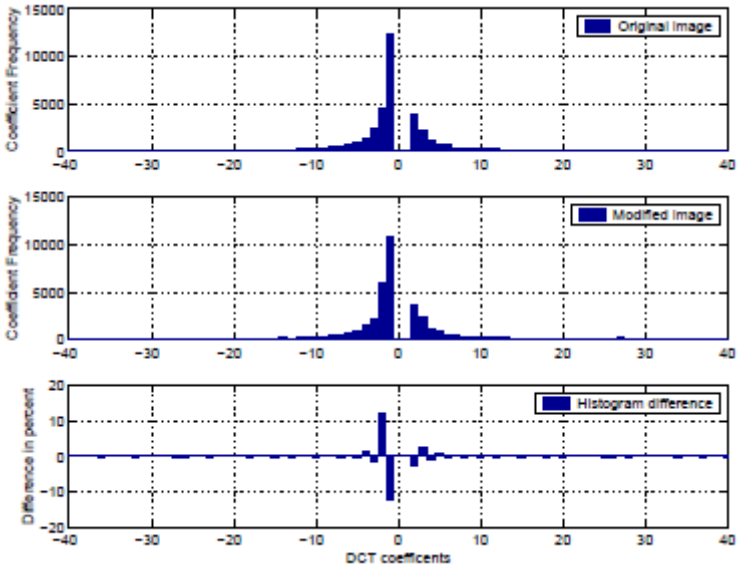
3.Ukrywanie informacji w obrazach JPEG

Obrazy JPEG są powszechnie używane w Internecie, an stronach WWW. Tu wyjaśnimy krótko format JPEG i jak może być używany do ukrywania wiadomości. Format obrazu JPEG używa dyskretnej transformacji cosinusowej (DCT) do tarnsformacji bloków 8x8 pikseli obrau na 64 DCT współczynnika każdy. Mniej znaczące bity kwantowanego współczynnika DCT są wykorzystywane jako bity nadmiarowe w których osadzana jest ukrywana informacja. W niektórych formatach graficznych, np. GIF, wizualna struktura obrazu istnieje do pewnego we wszystkich warstwach bitowych obrazu. Systemy steganograficzne , które modyfikują najmniej znaczących bitów obrazów w tych formatach są często podatne na ataki wizualne. Nie jest to prawdą dla formatu JPEG. Modyfikacja pojedynczego współczynnika DCT wpływa na wszystkie 64 piksele obrazu. Z tego powodu, nei są znane ataki wizualne przeciwko obrazom formatu JPEG. Poniższe zdjęcia prezentują dwa obrazki o rozdzielczości 800x600 i głębokości koloru 24 bitów. Nieskompresowany orygianlny obraz ma rozmiar prawie 12 Mbm podczas gdy te dwa obrazy mają około 0.3Mb. Ten z lewej strony nie jest zmodyfikowany. Zdjęcie z prawej zawiera pierwszy rozdział "Polowania na Snarka" Lewisa Carrolla. Po kompresji, rozdział ma rozmiar około 14 700 bitów. Nie jest możliwe dla ludzkiego oka aby znaleźć wizualnie różnicę między nimi dwoma

4.Analiza statystyczna

Testy statystyczne moga ujawnić czy obraz był modyfikowany przez steganografię, przez przetestowanie czy statystyczne właściwości obrazu odbiegają od normy. Niektóre testy są niezależne od formatu danych i mierza tylko entropię danych nadmiarowych.. Najprostszy test mierzy korelację w przód. Bardziej złożonym jest "Universal Statistical Test for Random Bit Generators" Ueli Maurera. Oczekujemy ,że obrazek z ukrytymi danymi ma wyższą entropię niże ten bez. Te proste testy nie mogą automatycznie decydować czy obraz zawiera ukrytą informację. Westfeld i Pfitzman zaobserwowali ,że osadzona zakodowana dana w obrazie GIF zmienia historgma częstotliwości koloru. Jedną z właściwości danej zaszyfrowanej jest to ,że bit jeden i zero są prawdopodobnie równe. Kiedy używamy metody najmniej znaczących bitów do osadzania zakodowanej danej w obrazie, który zawiera kolor dwa częściej niż kolor trzy, kolor dwa jest zmieniany częściej niz kolor trzy niż na odwrót. W rezultacie, różnica w częstotliwości kolorów miedzy dwa a trzy została zredukowana przez osadzanie.Jest top prawdziwe dla obrazów JPEG. Zamiast mierzenia częstotliwości kolorów, analizujemy częstotliwość współczynnika DCT. Rysunek pokazuje przykład , w którym, osadzenie ukrytych wiadomości powoduje zauważalne

różnice w stosunku do współczynnika DCT histogramu



Użyjemy testu X^2 dla określenia czy obrazek pokazuje odkształcenia osadzonych ukrytych danych. Ponieważ test uzywam medium steganograficznego, oczekiwany rozkład y_i^* testu X^2 musi być wyliczony z obrazka. Niech n_i będzie częstotliwością współczynnika DCT w obrazie. Zakładamy ,że obrazek z osadzonymi ukrytymi danymi ma podobną częstotliwość co sąsiedni współczynnik DCT. W wyniku możemy przyjąć średnią arytmetyczną

$$y_i^* = \frac{n_{2i} + n_{2i+1}}{2},$$

dla określenia oczekiwanego rozkładu. Oczekwiany rozkład jest porównywany z zaosberwowanym

$$y_i = n_{2i}.$$

Wartość X^2 dla różnic między rozkładami jest dany

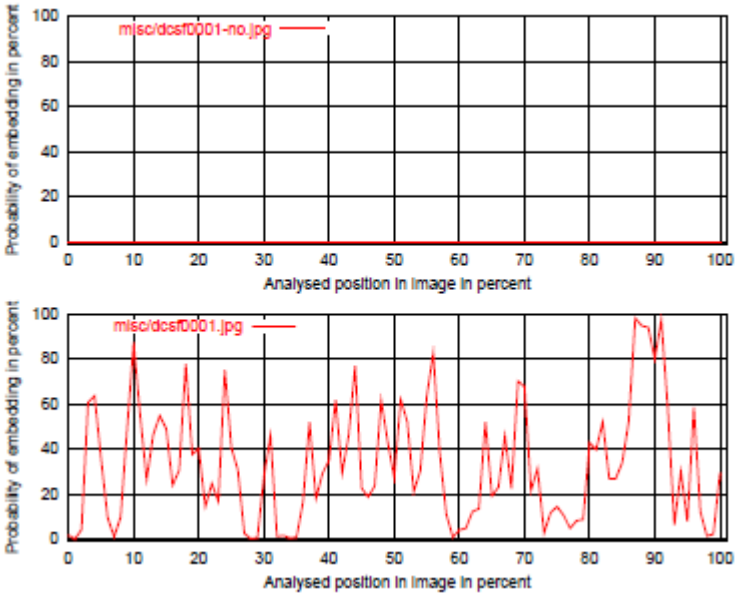
$$\chi^2 = \sum_{i=1}^{\nu+1} \frac{(y_i - y_i^*)^2}{y_i^*}$$

gdzie v to stopnie swobody, to znaczy ,liczba różnych kategorii w histogramie minus jeden. Prawdopodobieństwo osadzenia p jest wtedy dane przez funkcję dopełnienia dystrybuanty

$$p = 1 - \int_0^{\chi^2} \frac{t^{(\nu-2)/2} e^{-t/2}}{2^{\nu/2}\Gamma(\nu/2)} dt,$$

gdzie Γ jest to funkcją gamma Eulera

Możemy obliczyć prawdopodobieństwo osadzenia różnych części obrazka. Selekcja zależy od tego jaki system steganograficzny chcemy wykryć. Dla obrazków, które nie zawierają ukrytej informacji, oczekujemy prawdopodobieństwa osadzenia na zero. Poniższy rysunek pokazuje prawdopodobieństwo osadzenia dla obrazka be zawartości steganograficznej i dla obrazka, który ma w sobie ukrytą zawartość



5.Systemy steganograficzne

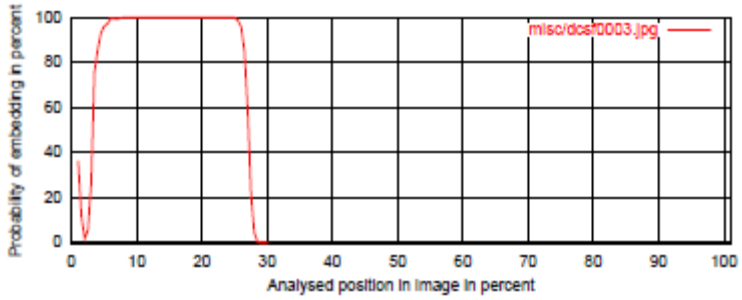
Zaprezentujemy tu kilka systemów steganograficznych, które osadzają ukryte wiadomości w obrazach JPEG. Pokażemy ,że zakłócenia statystyczne zależą od systemu steganograficznego, który wstawia wiadomość do obrazka. Ponieważ zakłócenia są charakterystyczne dla każdego systemu, stworzymy sygnaturę, która pozwoli nam zidentyfikować jaki system został zastosowany. Są trzy popularne systemy steganograficzne dostępne w Internecie, które ukrywają informacje w obrazkach JPEG:

- JSteg, Jsteg-Shell
- JPHide
- OutGuess

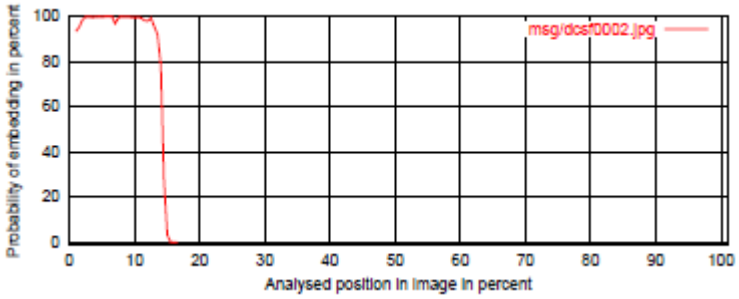
Wszystkie te systemy używają pewnej formy osadzania w najmniej zanczących bitach i są wykrywalne przez analizy statystyczne. Poniżej zaprezentujemy określone charakterystki tych systemów i pokażemy jak je wykrywać.

5.1 JSteg, Jsteg-Shell

JSteg jest dodatkiem Dereka Uphama do biblioteki oprogramowania JPEG. Współczynniki DCT są modyfikowane w sposób ciagły od początku obrazka. JSteg nie obsługuje szyfrowania i nie ma losowego wyboru bitów. Dane wiadomości są poprzedzone nagłówkiem zmiennego rozmiaru.Pierwsze pięć bitów w nagłówku wyraża rozmiar długości pola w bitach. Kolejne bity zawierają długość pola, które wyraża rozmiar osadzonej wiadomości. Ponizszy rysunek pokazuje wynik testu X^2 dla obrazka zaawierającego informacje ukryte JSteg. W tym przypadku, pierwszy rodział "Polowania na Starka" zosytał skompresowany bzip2 przed osadzeniem.



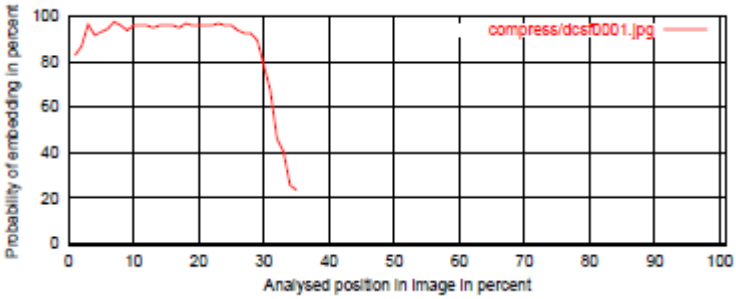
Niskie prawdopodoboieństwo na początku wykresu jest spowodowane prze słownik na początku pliku skompresowanego bzip2. Ten słownik nie wygląda na zaszyfrowaną daną i nei jest wykrywany przez test.. Jsteg-Shell jest interfesjem użytkownika Windows dla JSteg. Został zaprojektowany przez Korejwa i obsługujeszyfrowanie i kompresję zawartości przed osadzeniem danej w JSteg. Jsteg-Shell używa szyfru strumieniowego RC4 do szyfrowania. Jednak przestrzeń klucza RC4 jest ograniczona do 40 bitów. Kiedy stosowane jest szyfrowanie, oczekujemy ,że prawdopodobieństwo osadzenia będzie wyższe na początku obrazka. Nie powinno być wyjątków. Przykład zastosowania Jsteg-Shell jest pokazany poniżej



Obserwacja wykresu pozwala nam określić rozmiar osadzonej wiadomości. Później pokażey jak można poprawić to aby móc automatycznie wykrywać zawartość steganograficzną

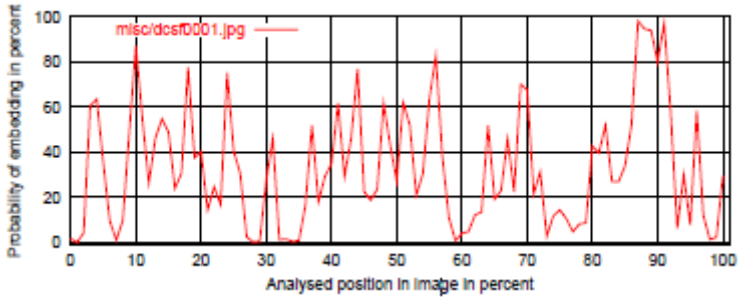
5.2 JPHide

JPHide jest systemem steganograficznym Allana Lathama. Sądwie wersje 0.3 i 0.5. Wersja 0.5 obsługuje dodatkową kompresją ukrytej wiadomości. W wyniku, używa różnych nagłóków do przechowywania osadzonej informacji. Przed osadzeniem zawartości, mamy szyfrowanie Blowfish z dostarczonym przez użytkownika hasłem. Ponieważ współczynniki DCT nie są wybierane w sposób stały od początku, JPHide jest trochę trudniejszy do wykrycia. Program używa stałej tabeli, która określa jaki współczynnik jest moyfikowany następny. Współczynniki są wybierane przez tablicę w taki sposób ,że współczynniki które mogą być liczbowo wysokei są używane jako pierwsze. Generator liczb pseudolosowych określa czy współczynniki są przeskakiwane. Prawdopodoboieństwo przeskoczenia bitów zależy od długości ukrytej informacji i tego jak wiele bitów już osadzono. JPHide nie tylko modyfikuje najmniej znaczące bity współczynnika DCT, może rónież przełączać do trybu gdzie drugi mniej znaczące bity są modyfikowane. Poniższy rysunek pokazuje prawdopodobieństwo osadzenia dla obrazka zawierającego ukrytą wiadomość z JPHide. Ponieważ JPHide może przeskoczyć współczynniki DCT, prawdopodobieństwo nie jest tak wysokie jak w JSteg.



5.3 OutGuess

OutGuess jest systemem steganograficznym dostępnym jako kod źródłowy UNIX. Są dwie wersje : OutGuess 013b , słaby pod kątem analizy statystycznej i OutGuess 0.2, który zawiera możliwości zabezpieczenia właściwości statystycznych i nie może być wykryty przez testy statystyczne używane tu przez nas. OutGuess różni się od powyższych systemów, tym ,że wybiera współczynniki DCT z generatora liczb pseudo losowych. Dostarczone przez użytkownika hasło inicjuje szyfr strumieniowy i generator liczb pseudolosowych, oba oparte o RC4. Szyfr strumieniowy jest używany do szyfrowania zawartości. Ponieważ modyfikacje są rozproszone losowo przez współczynniki DCT, test X^2 nie może być zastosowany w stale zwiększanym próbkowaniu obrazka. Zamiast tego przesuwamy się do pozycji gdzie pobieramy próbkę obrazu.
Przy OutGuest 0.13 nie musimy znajdować żadnej wyraźnej sygnatury. Poniższy rysunek pokazuje prawdopodobieństwo osadzenia dla próbki obrazka. Igły wskazują obszary w obrazku gdzue modyfikacja współczynników powoduje odchylenia od oczekiwanej częstotliwości współczynnika DCT.



6.Środowisko wykrywania

Wcześniej przedstawiliśmy sygnatury wykrywania, które pozwalają nam znajdować ukryte wiadomości i określać jaki system steganograficzny został użyty do ich osadzania. Teraz zaprezentujemy "Stegdetect", zautoamtyzowane narzędzie do analizy obarazów JPEG z zawartością steganograficzną.

6.1 Stegdetect

Stegdetect wykrywa obrazy , które mają zawartość ukrytą JSteg, JPHide i OutGuess 0.13b. Dla kazego systemu , który chcemy wykryć, wybieramy współczynnik DCT w porządku w którym są modyfikowane i stosujmy test X^2. Dane wyjściowe ze Stegdetect wyświetlają systemy steganograficzne zanlezione w obarzku lub "negative" jeśli nic nie znaleziono Stegdetect wyraża poziom zaufania detekcji od jednej do trzech gwaizdek. Poniżej mamy przykładowe dane wyjściowe:

```
misc/0003-wonder-2.jpg : jphide(*)
misc/dscf0001.jpg : outguess(old)(***)
misc/dscf0002.jpg : negative
misc/dscf0003.jpg : jsteg(***)
```

6.1.1 Wykrywanie JSteg

JSteg nie modyfikuje współczynnika DCT zerem i jedynką. Z tego powodu, są one ignorowane w teście X^2 . Próbkujemy współczynniki DCT od początku obrazka i obliczyć prawdopodobieństow osadzenia. Proces ten jest powtarzany ze wzrostem rozmiaru próbki dopóki współczynniki DCT są zawarte w próbce. W optymalizacji wydajności, zatrzymujemy obliczanie prawdopodobieństwa osadzania kiedy spadnie poniżej pewnego progu. Aby poprawić dokładność wykrywania,szacujemy wielkość ukrytej zawartości z wyliczonego wykresu i porównujemy ją z rozmiarem przechowywanym w osadzonym nagłówku JSteg.

6.1.2 Wykrywanie JPHide

Ponieważ JPHide modyfikuje współczynniki DCT w stałym porządku przez tabelę, zmieniamy współczynniki w celu obliczenia prawdopodobieństwa osadzenia. Jednak są dwa wyjątki, które wpływają na wykrywanie. JPHide modyfikuje współczynnik -1,0 i 1 w specjalny sposób. W wyniku, modyfikacja tych współćzynników nie będzie wykryta przez test X^2. Jednak, proste zignorowanie tych wpsółczynników pozwala nam jeszcze wykryć zawartość osadzoną JPHide. Również ingnorujemy modyfkację drugieg mniej zanczącego bitu. Podobnie jak JSteg, zatrzymujemy obliczanie prawdopodobieństwa osadzenia, kiedyspadnie poniżej pewnego progu.

6.1.2 Wykrywanie OutGuess

Wykrywanie zawartości osadzoen OutGuess 0.13b jest złożone przez fakt ,że współczynniki sa wybierane pseudolosowo,nie ma usatalonej kolejności w jaksie stosuje się test X^2. Jednak Provos pokazał ,że test X2 może być rozszerzony do wykrywania ukrytej zawartości w OutGuess 0.13b. Zamiast zwiększania rozmiaru próbki, używamy stałęgo rozmairu próbki, ale przesuwamy do poozycji gdzie próbki są pobierane w całym zakresie obrazka. Test zaczyna się na początku obrazka, a pozycja jest zwiększana o jeden procent dla każdego zastosowania testu X^2. Test rozszerzony nie reaguje na niezmodyfikowany obraz,. Ale wykrywa osadzanie w pewnym obszarze obrazu steganograficznego. Aby znaleźć właściwy rozmiar próbki, wybieramy oczekiwany rozkład dla rozszerzonego testu X2, który powinien spowodować negatywny wynik testu .Zamiast obliczania arytmetycznego, bierzemy średnią arytmetyczną dwóch niepowiązanych współczynników:

$$y_i^* = \frac{n_{2i-1} + n_{2i}}{2}.$$

Binarne wyszukiwanie rozmairuy próbki jest używane do znajdowania wartości dla której rozszerzony test X^2 nie pokazuje korelacji z oczekiwanym rozkładem pochodzącym z niezależnych współczynników.

6.1.4 Wydajność Stegdetect

Przeanalizujemy wydajność Stegdetect na prcesorze Celeron 333 Mhz przez pomiar czasu w jakim przetwarza kilkaset plików JPEG. Wynik jest średnią liczbą kilobajtów, któe mogą być przetwaorzone na sekundę (Kbps). Przetestujemy wydajność oddzielnie dla każdego systemu stgeanograficznego, a potem zmierzymy wydajność dla wszystkich testów wspólnie. Wyniki są pokazane na poniższym rysunku . Test JSteg jest najszybszy a wykrywanie JPHide i OutGuess 0.13b jest mniej więcej równe

Test	Speed
JSteg	356 KBps
JPHide	200 KBps
OutGuess 0.13b	227 KBps
All tests	127 KBps

Podając wyniki dla oddzielnych testów, można było oczekiwać połączenia szybkości dla wszystkich testów, o około 80 Kbps. Jednak szybkość jest wyższa ponieważ testy dla JPHide i OutGuess 0.13b są pomijane jeśli wykryto JSteg. Dla skalibrowania czułości wyklrywania Stegdetect, przetestowaliśmy około 1500 obrazków robionych aparatem Fuji. Wyniki są pokazane na poniższym rysunku

Test	False Negatives
JSteg	2%
JPHide	15% – 60%
OutGuess 0.13b	60%

Procent fałszywych negative zależy od systemu steganograficznego i rozmiaruy osadzonej informacji. Mniejsza wiadomość, tym trudniej jest wykryć średnią statystyczną. Stegdetect jest bardzo wiarygodny w znajdowaniu obrazków, które mają osadzoną wiadmość z JSteg. Dla naszych przykładowych obrazków , znalzło się tylko około 2% negatives. Dla JPHide między 15% a 60%. JPHide 0.5 jest trudniejszy do wykrycia ponieważ kompresuje zawartość przed osadzeniem. Współczynnik dla OutGuess 0.13b to około 60%.

6.2 Znajdowanie obrazków

Teraz kiedy możemy automatycznie przetestować zawartość steganograficzną, jesteśmy gotowi na wysuzkiwaniea obrazków, ,które mogą mieć osadzana ukrytą wiadomość. Oczywistym położeniem dla wyszukiwania takich obrazków są strony w Internecie. Robot internetowy , który znajduje obrazki JPEG może dostarczyć Stegdetect wystarczająco danych. Stworzyliśmy "Crawl", prosty ale efektywny robot internetowy, który zapisuje obrazki JPEG ,które napotyka na stronach WWW. Użyliśmy "libevent", biblkioteki dla asynchronicznego zapisywania zdarzeń, Crawl jest zaimplementowany w mniej niż 5000 linii kodu źródłowego C. Crawl dokonuje głebokiego wyszukiwania i ma następujące cechy:

- Obrazki i strony www mogą być dopasowane wyrażeniamio regularnymi. Dopasowanie moze być użyte do dołączenia lub wykluczenia strony z wyszukiwania
- Można określić minimalny i maksymalny rozmia obrazka. Pozwala to nam wyodrębniać obrazki, które są zbyt małe do ukrycia informacji. Ogrniczyliśmy rozmiar obrazków w zakresie od 20 KB do 400 KB.
- Żądania DNS są synchroniczne ale buforowane. Synchroniczne zapytania DNS mogą być główną przeszkodą wydajności ponieważ powodują one blokowanie robota.

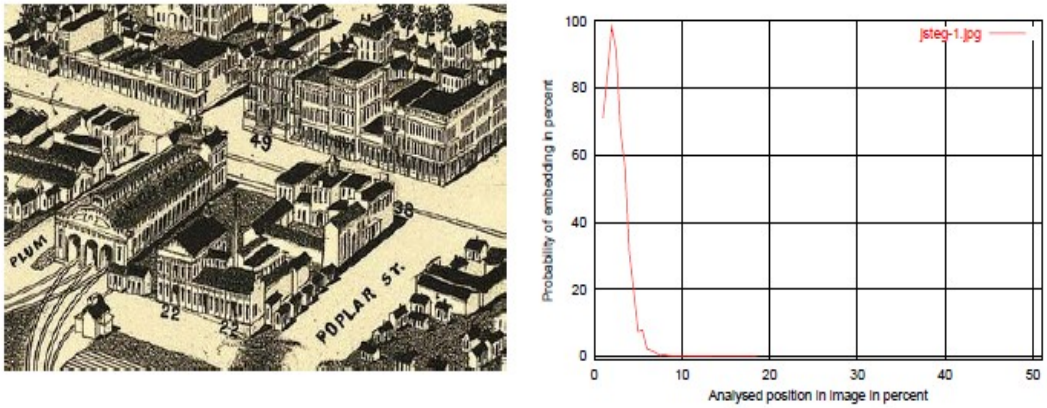
Poniższy rysunek pokazuje dane wyjściowe z Crawl

```
HEAD http://img.andale.com/635/monitor_lo.jpg
HEAD http://img.andale.com/635/hi.jpg
GET http://www.cities.com/a_ports/graphone.jpg
GET http://img.andale.com/635/scope_lo.jpg
Terminated with 3479 saved urls.
448684 GET for body 2861924 Kbytes
436084 HEAD for header 271287 Kbytes
   9.172 Requests/sec
```

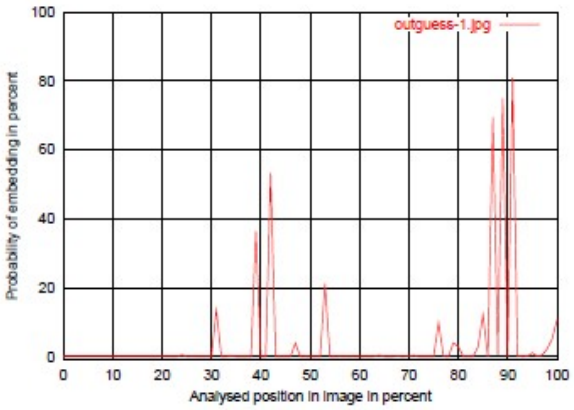
Dla zautomatyzowania wykrywania Crawl używa "stdout" . Ponieważ Stegdetect może zaakceptować obrazki ze "stdin", połączyliśmy Crawl z Stegdetect przez potkowanie dla zautomatyzowania procesu wykrywania. Po przetworzeniu dwóch milionów obrzków Stegdetect, odkryliśmy ,że pona 1% wszystkich obrazków wydawało się zawierać ukrytą zawartość.JPHide był wykrywany najczęściej.

Test	False Positives
JSteg	0.003%
JPHide	1%
OutGuess 0.13b	0.1%

Większość z nich jest fałszywie dodatnia. Axelsson zastosował "Base-Rate Fallacy" do systemów wykrywanai włamań i pokazał ,że wysoki procent fałszywie dodatnich ma znaczący wpływ na efektywność takiego systemu. Sytaucja jest bardzo podobna dla Stegdetect. Bezpiecznie zakłada ,że procent obrazków zawierajacych steganograficzną treść jest mniejsza w porównaniu z procentem fałszywie dodatnich. Jako wynik,"prawdziwie dodatni" wskaźnik, tj. prawdopodobieństwo ,ze brazek wykryty przez Stegdetect rzeczywiście ma treść steganograficzną, ma wpływ gównia przez współczynnik fałszywie dodatnich. Odnotujmy, że są specjalne kalsy obrazków dla których Stegdetect fałszywie wskazuje ukrytą zawartość. Przykład fałszywie dodatniego jest pokazany na poniższym rysunku



Stegdetect wskazuje ,że zawartość został ukryta przez JSteg. Jednak , kiedy analizujemy prawrdopodobieństwo oszadzenia wyświetlone obok rysunku,nie widzimy płaskowyżu na początku , jak można by oczekiwać dla osadzonych danych zaszyfrowanych. Podobnie znajdujemy fałszywie dodatnie kiedy próbujemy wykryć ukrytą treść OutGuess. Obrazki z monotonnym tłem,jak poniżej



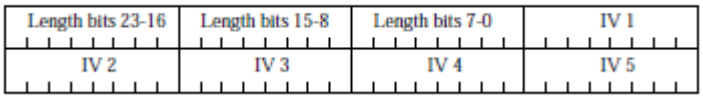
są bardziej prawdopodobne ,że są fałszywie dodatnie. Kiedy analizuejmy wykres, widać tylko ,że kilka szpilek wysokiego prawdopodobieństwa. Jeśli byałby ukryta zawartość, moglibyśmy oczekiwać znalezienia więcej obszarów w obrazie , gdzie rozserzony test X^2 pokazuje dodatnie wyniki. To ,ze Stegdetect znajduje tak wiele obrazó, które wydają się mieć ukryte treści przez JPHide, nie oznacza ,że jest tak wiele obrazków, które naprawdę zawierają ukryte treści. Oznacza to ,że funkcje wykrywania JPHide należ poprawić aby były bardziej precyzyjne. Ponadto wiele zdjęć pobranych z Internetu jest bardzo niskiej jakości, a obrazy uzywane do kalibrowania Stegdetect są wyższej jakości, ponieważ pochodza z kamery cyfrowej.

6.3 Weryfikacja ukrytej zawartości.

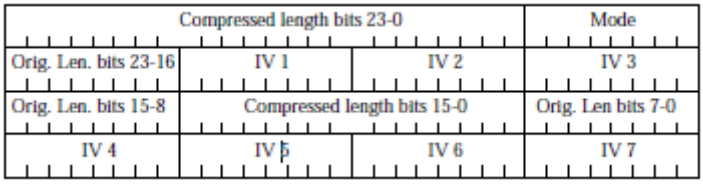
Testy statystyczne używane do znajdowania treści steganograficznych w obrazach , wskazują nic więcej jak to ,że zawartośćzostała osadzona. Z tego powodu, Stegdetect nie może zagwarantować istnienie ukrytej wiadamości. Aby zweryfikować ,że obraek ma ukrytą zawartość, konieczne jest uruchomienie ataku słownikowego na plik JPEG. Stegbreak robi to dla zawartości Jsteg-Shell, JPHide lub OutGuess 0.13b. Ponieważ wszystkie prezentowane systemy steganograficzne ukrywają zawartość w oparciu o hasło użytkownika, atakujący może próbować odgadnąć jhasło określajace jaka zawartość została ukryta. Zamiast próbowania wszystkich możliwych haseł, dużo szybciej jest wypróbować tylk osłowa ze słownika, tj. ataku słownikowego. Dla takiego ataku, konieczne jest aby użytkownik systemu steganograficznego wybrał słabe hasło z małego podzbioru pełnej przestrzeni haseł. Atak kluczem na system kryptograficzny często ma tą korzyść ,że właściwości podsatwowego tekstu jawnego są znane atakującemu. Przy danych tych właściwościach, możliwe jest statystyczne zweryfikowanie czy został znaleziony poprawny klucz deszyfrujący. Wszystkie te systemy steganograficzne osadzają nagłówek informacji dodatkowo w wiadomości w obrazku. Informacje nagłówkowe zawierają oprócz innych rzeczy, długość ukrywanej treści. Możemy użyć tej informacji do zweryfikowania poprawności odgadywanego hasła.

6.3.1 Nagłówek informacji JPHide

JPHide 0.3 osadza 64 bitowy nagłówek. Pierwsze 24 bity onbejmują długość ukrytej wiadomości w bajtach. Pozostałe 40 bitów jest uzyskiwanych z szyfrowania pierwszych ośmiu współczynników DCT z Blowfish. Generowanie klucza Blowfish jest inicjalizowane odgadywanym hasłem. JPHide pobiera pierwsze osiem współczynników DCT, redukując jes modulo 256 a potem łączy aby uzyskać 64 bitowy blok. Blok ten jest potem szyfrowany, a pierwsze 3 bajty są nadpisywany informacją o długości. Wynik jest pokazany na poniższym rysunku



Atak słownikowy używa 40 bitów IV jako weryfikatora. Dodatkowo, możemy sprawdzić czy kodowana długość wypełnia obrazek.
Nagłówek dla JPHide 0.5 jest dwukrotnie dłuższy niż JPHide 0.3; ponieważ JPHide 0.5 kompresuje wiadomość przed osadzeniem, nagłówek zawiera zarówno skompresowaną jak i pierwotną długość wiadomości. Ze zwiększeniem długości nagłówka, uzyskujemy 56 bitowy weryfikator. IV jest uzyskiwane przez szyfrowanie pierwszych 16 współczyników DCT,i jest potem nadpisywany informacją o długości. Dodatkowo, do 56 bitów, uzyskujemy również 16 bitów więcej dla weryfikacji naszego odgadywania ponieważ część skompresowanej długości została zduplikowana w nagłówku



Inną różnicą między wersją 0.3 i 0.5 jest zmiana obliczenia generowanego klucza . W wersji 0.5 , generowanie klucza Blowfish zależy od pierwszy ośmiu współczynników DCT. W wyniku, wygenerowany klucz Blowfish musi być obliczony ponownie dla obrazków które różnią się tymi współczynnikami DCT. Powoduje to ,znaczne spowolnienie Stegbreak

6.3.2 Informacje nagłówkowe Jsteg-Shell

Jsteg-Shell jest bardzo prosty. Ponieważ jest tylko interfejsem użytkownika dla JSteg, nie szyfruje długości osadzonej wiadomości. Zamiast tego dodaje sygnaturę na końcu wiadomości. Sygnatura to albo "korejwa", "cMk4" lub "cMk5". Uzyskujemy 32 bitową pewność,że odgadliśmy poprawne hasło. Jednak , ponieważ rozmiar klucza jest ograniczony do 40 bitów, możliwe jest przeszukanie całej przestrzeni klucza zamiast użyć ataku słownikowego

6.3.3 Informacje nagłówkowe OutGuess

Ataki słownikowe przeciwko OutGuess wydają się być niemożliwe, ponieważ brakuje informacj dla zweryfikowania odgadniętego hasła. OutGuess przechowuje 32 bitowy nagłówek przed osadzoną wiadomością. Nagłówek składa się z 16 bitowego początku i 16 bitów zawiwerających długość wiadomości w bajtach. Możemy użyć tylko długośći dla weryfikacji naszego odgadniętego hasła, ponieważ początek może być dowolną liczbą. Jako dodatkowe sprawdzenie, Stegbreak pobiera 512 bajtów zaszyfrowanej wiadomości i sprawdza uzyskane bajty dla losowości. Najprostszym i najszybszym sprawdzeniem jest zliczenie ilości bitów zer i jedynek. Jeśli jest blisko 50% bitów jeden wtedy dana wydaje się prawdopodobnie losowa. Dalej zwiększamy dokładność przez zastosowanie prostego tetsu statystycznego dla danej. Jednak 512 bajtów danych nie jest wystarczające na przejście testu. Dla dużego słownika , jeszcze znajdujemy zbyt wielu kandydatów na hasło, tworząc niewykonalnym atak słownikowy.

6.3.4 Wydajność Stegbreak

Pomiary Stegbreak dokonane zostały na Pentium III 1200 Mhz. Wyniki są pokazane na poniższym rysunku

System	Speed
JPHide	15,000 words/s
OutGuess 0.13b	47,000 words/s
JSteg	112,000 words/s

Dla JPHide, możemy sprawdzić do 15000 słów na sekundę. Test został uruchomiony z 300 obrazkami i słownikiem z około 577 000 słów. Stegbreak jest wolnyu ponieważ musi sprawdzić obiee wersje JPHide. Kiedy sprawdza wersję 0.5, generowany klucz Blowfish musi być wyliczany ponownie dla prawie każdego obrazka. Stegbreak jest szybszy dla OutGuest: może sprawdzić około 47 000 słów na sekundę. Jednak, przy dużym słowniku, narzędzie znajduje kandydatów na hasła dla każdego obrazka. Dla Jsteg-Shell, możemy sprawdzić około 112 000 słów na sekundę. To jest dośćszybko aby uruchomić atak słownikowy na pojedynczym komputerze. Jednak ponieważ przestrzeń klucza jest ograniczona do 40 bitów, ma sens wykoannei aktaku brute-force dla całej przestrzeni klucza. Przesztrzeń klucza jest redukowana do 40 bitów w taki sposób ,że efektywnie jest wykorzystywane tylko 35 bitów.

6.4 Rozproszony atak słownikowy

Jak widać, Stegbreak jest zbyt wolny aby uruchomić atak słownikowy przeciwko JPHide na pojedynczym komputerze. Jednakże, ponieważ atak słownikowy jest z natury równoległy, możliwe jest rozproszenie ataku słownikowego na wiele stacji roboczych. Takie rozproszone środowisko obliczeniowe powinno działać w kalstrze luźnych stacji roboczych spełaniających następujące wymagania:

- Konfiguracja i zarządzanie powinny być proste
- Powinno być przenośne na wiele systemów operacyjnych, tak ,że można użyć wielu różnych systemów komputerowych.
- Cała komunikacja powinna byćszyfrowana i uwierzytelniana
- System nie powienien wymagać uprzywilejowania "roota" dla instalacji

Ponieważ tyaki system nei jest dostępny jako open-source, stworzyliśm "Disconcert". Disconcert używa libevent dla asynchronicznego powiadamiania o zdarzeniach, i "libio", bibliotekę spejalnie stworzoną do użycia z Disconcert" . Libio dzieli komunikację na źródło danych i odbió danych. Żródło danych jest połączone odbiorema danych przez wiele filtrów. Poniżej wyjaśnimy kilka poleceń Disconcert:

- Poleceń init przenosi pliki do wybranego klienta. Jest używane do kopiowania Stegbreak, listy słów i plików obrazków do zdalnego komputera.
- Polecenie job ustawia różne parametry dla określonej pracy. Zawiera to liczbę jednostek pracy, które powinny być skończone a linia poleceń będzie wykonywać się na maszynach klientów
- Polecenie run jest używany do startowania zdalnego wykonania pracy. Disconert ustawia poziom "miły" dla tych prac na 10 więc nie drażni użytkowników stacji roboczych.

Klienci wysyłają status wyjścia zakończonego procesu do serwera aby wskazać czy jednostka pracy zakończyła się sukcesem czy nie. Komunikuje odgadnięte hasło lub inną wiadomość do serwera, "stdout" i "stderr" są przekierowywane do plików na serwerze.

Jeśli klient utraci swoje połączenie z serwerem cała komunikacja jest buforowana dopóki klient nie odzyska połączenia. Jeśli klient nie uzyska połączenia wewnątrz określonego czasu ramki, serwwer przynza jednostkę pracy tego klienta innej maszynie. Środowisko Disconcert również obsługuje wiele prac w tym samym czasie. Aby zabezpieczyć niedozwoloną zawartość (taką jak pornogafia) przed klientem, Stegbreak może wyodrębnić tą informację z obrazków JPEG , która jest odniesiona do ataku słownikowego i zapisana w oddzielnym pliku. Dla JPHide, atak słownikowy wymaga tylko około 512 bajtów dla zweryfikowania odgadywanego hasła. Inną korzyścią tego jest redukcja

ruchu sieciowego. Stegbreak ma bardzo niskie I/O i wymaganą pamięć i jest trudno odczuwalny kiedy jest uruchamiany w tle. Całkowita wydajność kiedy próbujemy znaleźć zawartość ukrytą przez JPHide to około 200 000 słów na sekundę. Jest to 15 razy szubsze niż uruchamianie na pojedynczym Pentium III 1200 Mhz. Najwolniejszy klient obrabia 471 słów na sekundę, najszybszy 12 504 słów na sekundę. Średnia wydajność stacji roboczej to około 3900 słów na sekundę.

7.Omówienie

Crawl pobrał ponad dwa miliony obrazków z aukcji eBay. Spośród nich, Stegdetect wskazał ,że około 17 000 ma zawartość steganograficzną. Z tych 17 000, 15000 przypuszczalnie miało zawartość ukrytą przez JPHide. Wszystkie 15000 obrazków zostało przetworzonych przez Stegbreak. Stegbreak uruchomionu był w klastrze 60 maszyn, ale i tak był zbyt wolny aby przetworzyć wszystkie obrazki jakei znalazł Stegdetect. Aby sprawdzić poprawność wszystkich uczestniczących klientów, możemyu wstawiść śkedzenie obrazków dla każdej pracy Stegbreak. Zgodnie z oczekiwaniami, atak słownikowy znalazł poprawne hasło dla tych obrazków. Jednak ,jak dotąd nie znaleźliśmy żadnej pojedynczej ukrytej wiadomości. Oferujemy trzy wyjaśnienia wsparcia naszych wyników:

- Steganografia nie jest znacząco używana w Internecie
- Nikt nie używa systemów steganograficznych jakie możemy znaleźć
- Wszyscy użytkownicy systemów steganograficznych ostrożnie dobierają hasła, które nie są podatne na ataki słownikowe

Nawet jeśli większość haseł używanych do ukrywania treści było silnych, będzie mały procent haseł słabych. Słabe hasła są podatne na atak słownikowy i powinniśmy je móc znaleźć. Podobnie, nawet jeśli większość systemów steganograficznych używanych do ukrywania wiadomości była nierwykrywalna przez nasze mmetody, powiniśmy znaleźć obrazki z ukrytymi wiadomościami dla systemów wykrywalnych