Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42.056
DOI: https://doi.org/10.24867/06BE47Misic

STEGOSPLOIT U SISTEMIMA ZA PRENOS I PRIKAZ DIGITALNIH SLIKA

STEGOSPLOIT IN SYSTEMS FOR TRANSFERING AND DISPLAYING IMAGES

Aleksandra Mišić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – Primenjeno softversko inženjerstvo

Kratak sadržaj – *U ovom radu predstavljena je vrsta napada pod nazivom stegosploit. Opisan je postupak kreiranja malicioznog fajla, načini za isporuku fajla žrtvi, kao i načini odbrane od ove klase kiber pretnji.*

Ključne reči: *stegosploit, slike, sigurnost, steganografija, JPEG, PNG, poliglot*

Abstract – *This paper presents the class of attacks called stegosploit. We describe how the malicious files are created and delivered, as well as possible security controls for mitigating this class of cyber threats.*

Keywords: *stegosploit, images, security, steganography, JPEG, PNG, polyglot*

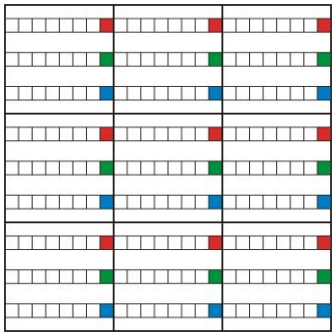
1. UVOD

Stegosploit je tehnika sakrivanja potencijalno malicioznog JavaScript koda u digitalne slike. Osmislio ju je istraživač sigurnosti u oblasti računarskih sistema, Saumil Shah i predstavio javnosti 2015. godine [1]. Naziv tehnike se zasniva na pojmu steganografija, koji označava skup tehnika za prikrivanje poruka unutar drugih poruka, slika, videa.

Stegosploit čine sledeći koraci:
1. Upisivanje koda u piksele slike uz pomoć odabranog steganografskog algoritma. U ovom radu, biće prikazan postupak za slike formata – *Joint Photographic Experts Group* (JPEG) i *Portable Network Graphics* (PNG);
2. Kreiranje poliglotskog fajla (eng. *polyglot*) koji u isto vreme predstavlja validan *Hypertext Markup Language* (HTML) dokument kao i validnu sliku;
3. Isporuka stegosploita žrtvi.

2. UPISIVANJE KODA U PIKSELE SLIKE

Uprkos postojanju sofisticiranog steganografskog algoritma, pod nazivom F5, koji čuva integritet kodirane poruke prilikom osnovnih transformacija slike, stegosploit koristi jednu od jednostavnijih tehnika (ili njenu varijaciju) – sakrivanje u najmanje značajne bitove (eng. *Blind Hide in Least Significant Bit* (LSB)) [2]. Razlog je potreba za konciznim i jednostavnim algoritmom, koji je u isto vreme sasvim dovoljno efikasan.
Digitalna slika se može predstaviti kao niz piksela. Svaki piksel sadrži najmanje tri kanala – crveni, plavi i zeleni (R, G, B). Neki formati slika (npr. PNG) podržavaju i četvrti – alfa kanal (A), koji se koristi za nivo transparencije piksela [3]. Ukoliko se za primer uzme slika čiji je svaki kanal predstavljen uz pomoć osam bitova, slika se može razložiti u osam bit-slojeva. Svaki bit-sloj je kreiran tako što se zadrže vrednosti bitova na odgovarajućoj poziciji u nizu, za svaki kanal, dok se ostali bitovi anuliraju. Na primer, za nulti nivo zadržaće se svi bitovi najmanjeg značaja (eng. *least significant bit*, LSB) (slika 1), dok će se za sedmi nivo zadržati svi bitovi najvišeg značaja (eng. *most significant bit*, MSB).



Slika 1. Nulti bit-sloj

Algoritam *Blind Hide in Least Significant Bit* podrazumeva da se poruka pretvori u niz bitova i upiše u nulti bit-sloj, u jedan proizvoljni kanal ili u sva tri. Nakon upisivanja poruke, potrebno je pretvoriti niz piksela u JPEG ili PNG format, gde dolazi do problema kompresije podataka.

2.1. JPEG i PNG kompresije

Kompresija podataka (eng. *data compression*) je čest postupak u računarstvu koji podrazumeva transformisanje digitalnih podataka u oblik koji sadrži manje bitova od polaznog, najčešće u svrhu smanjenja zauzeća memorije i komunikacionih kanala, kao i trajanja transmisije fajlova [4]. Iako postoji veliki broj različitih formata i algoritama za kompresiju podataka, može se napraviti opšta podela na algoritme bez gubitaka podataka (eng. *lossless compression*) i algoritme sa izvesnim gubicima (eng. *lossy compression*). Algoritmi bez gubitaka podataka komprimuju podatke na način koji ima inverzan postupak za tačnu rekonstrukciju originalnih podataka. Sa druge strane, prilikom kompresije uz pomoć algoritama sa gubicima, dolazi do nepovratnog gubitka dela informacija, što implicira da nema garancije da će dekomprimovan fajl biti identičan polaznom.

U kontekstu digitalnih slika, tipičan primer kompresije sa gubicima jeste JPEG, dok PNG implementira algoritam kompresije bez gubitaka. JPEG definiše niz postupaka za kompresiju sirovih piksela, međutim najpoznatiji i najviše

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor prof. dr Imre Lendak.

korišćen metod jeste onaj koji definiše *JPEG File Interchange Format* (JFIF). Mreža piksela se deli na podmreže dimenzija 8x8 piksela, a zatim se nad svakom mrežom izvršava diskretna kosinusna transformacija (eng. *Discrete Cosine Transform* (DCT)). Ovaj postupak prima parametar koji definiše željeni nivo kvaliteta kompresije. Što je veća vrednost ovog parametra, aproksimacija piksela je blaža, a gubitak podataka manji.

2.2. Algoritam zapisivanja koda u JPEG sliku

Da bi se izbegao problem pogrešnog dekodiranja koda iz JPEG slike prouzrokovan kompresijom sa gubicima, postupak enkodiranja i dekodiranja, kompresije i dekompresije se iterativno ponavljaju. Ukoliko se ovom procesu proslede odgovarajući parametri, greška se smanjuje nakon svake iteracije. Drugim rečima, broj različitih piksela između slika na početku i kraju iteracije, u pravilnim okolnostima, teži nuli.

Uvode se sledeće oznake:

S – originalna JPEG slika

K – kod za enkodovanje u sliku S

C - sirovi pikseli

ENCODE – steganografska funkcija za enkodovanje

DECODE – steganografska funkcija za dekodiranje

JPG – funkcija koja transformiše sirove piksele u JPEG sliku (JPEG kompresija)

INV_JPG – funkcija koja transformiše JPEG sliku u sirove piksele

b – bit-sloj (0-7)

c – kanal (R, G, B, svi)

g – mreža kodiranja

q – kvalitet JPEG kompresije [0, 1]

Pre početka prve iteracije, slika S se učitava u HTML element, *canvas,* i dobijaju se sirovi pikseli C (listing 1). U prvoj iteraciji se K upisuje u C uz pomoć ENCODE funkcije, a zatim se, uz pomoć JPG funkcije, postojeći pikseli kompresuju u JPEG sliku – S'. U nastavku prve iteracije, slika S' se učitava u *canvas* – C'' i dekodira se kod iz C'' uz pomoć DECODE funkcije.

```
C = INV_JPG(S)
C' = ENCODE(K, C, b, c, g)
S' = JPG(C', q)
C" = INV_JPG(S')
K' = DECODE(C", b, c, g)
```

Listing 1. Deo iteracije upisivanja koda u JPEG sliku

Dalje se proverava da li postoji razlika između K i K'. Ukoliko ne postoji, postupak enkodiranja je završen i konačan rezultat je S'. Ukoliko postoji, potrebno je zabeležiti trenutan broj različitih piksela između C' i C'', da bi se u sledećoj iteraciji proverilo da li se broj smanjuje, tj. da li teži nuli. Ukoliko ne teži, potrebno je izmeniti neke od parametara b, g i q.

Ukoliko je potrebna sledeća iteracija, postupak se ponavlja tako što se u C'' ponovo enkodira K.

2.3. Parametri funkcija enkodiranja i dekodiranja

Funkcije ENCODE i DECODE treba da budu svesne bit-nivoa (b), kanala (c) i mreže kodiranja (g) koji se koriste i ovi parametri moraju imati jednake vrednosti za obe funkcije.

Broj iteracija neophodan za uspešno enkodiranje zavisi od same biblioteke koja se koristi za JPEG kompresiju i dekompresiju (biblioteke su ugrađene u internet pretraživač). Mogu se sumirati faktori koji utiču na brzinu i verovatnoću konvergencije:

1. JPEG enkoderi primaju parametar kvaliteta kompresije (q) koji može imati vrednosti između 0 i 1. Vrednost 1 obezbeđuje najveći kvalitet krajnje slike i minimizaciju aproksimizacije, što ne garantuje kompresiju bez gubitaka, ali utiče na smanjenje broja iteracija i povećava verovatnoću konvergencije.

2. Ukoliko se bitovi koda ne upisuju u uzastopne bitove bit-sloja, već se preskače određen broj bitova u obe dimenzije (kvadratna mreža kodiranja - g), smanjuje se broj potrebnih iteracija. Pokazalo se da korišćenje mreža veličina 3x3 ili 4x4 donosi znatno bolje rezultate, dok dalje povećanje ovog broja ne pravi značajnu razliku.

3. Upisivanje koda u nulti ili prvi bit-sloj (b) najčešće ne dovodi do konvergencije. Slojevi 2 i 3 pokazuju bolje rezultate, dok je za više slojeve rezultat primetan za ljudsko oko.

2.4. Algoritam zapisivanja koda u PNG sliku

PNG slike koriste kompresiju bez gubitaka za skladištenje piksela. Iz tog razloga ne postoji problem gubitka steganografski enkodiranih poruka i nema potrebe za iterativnim postupkom opisanim u prethodnim poglavljima. I u ovom slučaju koristi se *canvas* i ugrađene funkcije PNG kompresije za kreiranje konačnog fajla.

Enkodovanje poruka u PNG slike ima određene prednosti u odnosu na enkodovanje poruka u JPEG slike:

1. Za proces enkodovanja, potrebna je jedna iteracija.

3. Moguće je enkodovanje u nulti bit-sloj, što ima najbolje vizuelne rezultate.

3. Enkodiranje i dekodiranje funkcioniše ispravno kada se koriste različiti pretraživači (eng. *cross browser encoding and decoding*).

3. KREIRANJE POLIGLOTSKOG FAJLA

U kontekstu računarskih sistema, poliglotski fajl predstavlja dokument koji zadovoljava strukturu dva ili više formata. Na primer, moguće je napraviti *Portable Document Format* (PDF) + ZIP poliglotski fajl koji se pravilno otvara u PDF čitaču kada mu je ekstenzija .pdf, odnosno uspešno eksportuje ako se ekstenzija promeni u .zip.

Stegosploit koristi HTML + JPEG/PNG poliglotske fajlove za isporuku napada žrtvi. Ukoliko se fajl otvori sa ekstenzijom .png ili .jpg, prikazaće se pravilna slika. Sa druge strane, ako se ekstenzija preimenuje u .html, u internet pretraživaču će se otvoriti validan HTML dokument.

3.1. JPEG/JFIF format

Kako bi se razjasnio postupak kreiranja HTML + JPEG poliglotskih fajlova, neophodno je upoznati se sa strukturom JPEG fajla, odnosno JPEG/JFIF formata. JFIF fajl se sastoji iz niza markera i marker segmenata. Svaki

marker se sastoji iz dva bajta od kojih prvi uvek ima heksadecimalnu vrednost FF, dok je drugi različit od FF i 00 i određuje tip markera [4]. Neki markeri su samostalni, dok većina označava početak marker segmenta koji ima strukturu kao na slici 2.

Marker	Segment	
FF xx	s1 s2	[Data bytes]

Slika 2. Struktura JFIF marker segmenta

Bajtovi s1 i s2 zajedno formiraju 16-bitni ceo broj koji označava dužinu *data bytes* dela, uključujući i dva bajta koje zauzima sam broj. Broj bajtova u nastavku iznosi: 265 * s1 * s2 – 2. U zavisnosti od tipa markera, *data bytes* sadrži različite specifične podatke.

Svaka JFIF slika obavezno počinje SOI (*Start of Image*) markerom, nakon kog sledi JFIF APP0 marker segment koji sadrži različite parametre slike (slika 3).

SOI	FF D8						
JFIF-APP0	FF E0	Dužina	JFIF\0				
		Verzija	U	Xres	Yres	W	H
DQT	FF DB						
…	…	ostali segmenti					

Slika 3. Početak JFIF slike

3.2. Kreiranje HTML + JPEG poliglotskog fajla

Da bi se napravio HTML + JPEG poliglotski fajl, neophodno je ubaciti HTML deo fajla na odgovarajuće mesto u JFIF strukturu sa slike 3. Kako APP0 marker segment ne utiče na renderovanje slike, već služi aplikacijama kao izvor metapodataka, najbolje mesto za upis jeste u ovaj segment nakon poslednjeg bajta. Iako nije neophodno, poželjno je ponovo podesiti vrednost polja za dužinu APP0 segmenta.

Na slici 4. se vidi sadržaj JPEG slike nakon upisivanja HTML koda. Plavom bojom obeležena su izmenjena polja.

SOI	FF D8						
JFIF-APP0	FF E0	Dužina	JFIF\0				
		Verzija	U	Xres	Yres	W	H
		<html>					
		<head> ...dekoder... </head>					
		<body> ...img... </body>					
		</html>					
DQT	FF DB						
…	…	ostali segmenti					

Slika 4. Poliglotski HTML + JPEG fajl

3.3. PNG format

Svaka PNG slika počinje osmobajtnim potpisom: 89 50 4E 47 0D 0A 1A 0A (heksadecimalno). Nakon toga sledi niz *Four Character Code* (FourCC) bajtova (eng. *chunk*) gde svaki ima sledeću strukturu:

1. Dužina: 4 bajta, pozitivan ceo broj, predstavlja dužinu podataka.

2. Tip: 4 bajta, primeri su IHDR, IDAT, IEND...

3. Podaci: varijabilna dužina.

4. *Cyclic Redundancy Check* (CRC): 4 bajta, predstavlja vrednost koja se generiše na osnovu polja – tip i podaci [5].

Svaki PNG fajl mora da sadrži tačno jedan IHDR segment i tačno jedan IEND segment. Između njih može biti proizvoljan broj IDAT segmenata koji sadrže podatke o pikselima (slika 5).

PNG Zaglavlje	89	50	4E	47	0D	0A	1A	0A	
IHDR	Dužina		IHDR		podaci segmenta				CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IEND	0		IEND		CRC				

Slika 5. Struktura PNG slike

3.4. Kreiranje HTML + PNG poliglotskog fajla

PNG fajlove je lakše proširiti nego JFIF fajlove, jer je dovoljno samo dodati još jedan segment u kom će se nalaziti HTML kod. Poseban tip segmenta je tEXt segment koji se može koristiti za skladištenje metapodataka o slici. HTML kod će biti upisan u jedan ovakav segment odmah ispod IHDR segmenta.

Na slici 6. se može videti PNG slika nakon što je u nju upisan HTML kod.

PNG Zaglavlje	89	50	4E	47	0D	0A	1A	0A	
IHDR	Dužina		IHDR		podaci segmenta				CRC
tEXt	Dužina		tEXt		<html>				
	<head> ...dekoder... </head>								
	<body> ...img... </body>								
	</html>								CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IDAT	Dužina		IDAT		piksel podaci				CRC
IEND	0		IEND		CRC				

Slika 6. Poliglotski HTML + PNG fajl

3.5. Dekodiranje i izvršavanje koda iz slike

Poliglotski fajlovi HTML + JPEG/PNG, dobijeni u prethodnom poglavlju, sadrže nekoliko ključnih komponenti:

1. sliku (JPEG ili PNG) sa steganografski enkodiranim JavaScript kodom;

2. JavaScript dekoder koji dekodira kod iz slike i izvršava ga;

3. HTML strukturu koja prikazuje samo sliku i pokreće dekoder po završetku učitavanja dokumenta.

Telo (eng. *body*) HTML dokumenta sadrži samo *img* element unutar *div* elementa:

```
<div>
        <img id=px src="#">
</div>
```

Listing 2. HTML deo poliglotskog fajla

Treba primetiti da je vrednost *src* atributa „#“, što znači da će vrednost izvora slike biti *Unified Resoure Locator* (URL) trenutnog dokumenta. Drugim rečima, slika koja sadrži enkodiran kod, koja je deo dokumenta, biće učitana u *img* element.

Po učitavanju dokumenta, pokreće se funkcija dekodera koja preuzima sliku iz *img* elementa i kreira *canvas*. U njega učitava piksele slike, a zatim briše originalni *img* element i na njegovo mesto postavlja pomenuti *canvas*. Kod se dekodira iz piksela *canvas*-a, inverznim

steganografskim postupkom, uz korišćenje odgovarajućih parametara koji se poklapaju sa vrednostima parametara enkodiranja (kanal, bit-sloj, veličina mreže enkodiranja). Od niza dekodiranih bitova, kreira se JavaScript string.

Najjednostavniji način da se dobijeni kod izvrši je uz pomoć eval() funkcije, međutim, ova funkcija se često obeležava kao potencijalno opasan kod i privlači neželjenu pažnju. Drugi način da se izvrši JavaScript string jeste da se kreira anoniman Function objekat i da se njegovom konstruktoru prosledi string kao argument (listing 3).

```
function execute(string) {
   var a = setTimeout(new Function(string), 100);
}
```

Listing 3. Funkcija za izvršenje malicioznog koda

Ukoliko se ovako konstruisan poliglotski fajl prikaže u pretraživaču kao .html fajl, pri vrhu prikaza će biti vidljiv tekstualni višak koji potiče od zaglavlja slike (slika 7).

Slika 7. Višak teksta na vrhu prikaza

Da bi se rešio ovaj problem, u fajl je dodat *Cascading Style Sheets* (CSS) deo kojim se višak teksta sakriva. Kako bi se maksimalno povećala verodostojnost poliglotskog fajla kada se otvara kao .html dokument, kreiran je dodatan CSS kod koji obezbeđuje da prikazi izgledaju identično kada se poliglotski fajl otvori u pretraživaču sa ekstenzijom .html i ekstenzijom .jpg ili .png (listing 4).

```
<style>
    body {
        background: #0e0e0e;
        color: transparent;
    }
    canvas {
        margin: auto;
        display: block;
        position: absolute;
        top: 0;
        bottom: 0;
        left: 0;
        right: 0;
    }
    img {
        visibility: hidden;
    }
</style>
```

Listing 4. CSS dodatak

CSS kod u ovom primeru je prilagođen često korišćenim internet pretraživačima iz listinga 5.

```
Google Chrome Version 76.0.3809.132
Firefox Quantum Version 68.0.2
Opera Version 62.0.3331.116
```

Listing 5. Internet pretraživači

3. DOSTAVLJANJE NAPADA ŽRTVI

Konačan poliglotski fajl moguće je dostaviti žrtvi na više načina. Jedna opcija je da napadač postavi fajl na server nad kojim ima kontrolu, a druga da postavi fajl na neki od sajtova koji služe za skladištenje fajlova. Primer ovakvog sajta je Filebin koji ne zahteva registraciju, dozvoljava *upload* HTML fajlova i pruža mogućnost kopiranja linka do dokumenta koji se otvara direktno u pretraživaču. Nakon što je postavio dokument na željeni sajt, napadač može da pošalje direktan link žrtvi uz pomoć nekog vida socijalnog inženjeringa. Napad se izvršava čim žrtva klikne na link.

4. NAČINI ODBRANE OD STEGOSPLOITA

Načini odbrane se mogu posmatrati iz ugla korisnika internet pretraživača i iz ugla programera koji kreiraju *web* aplikacije za *upload*-ovanje fajlova proizvoljnih formata. Korisnici treba da imaju svest da ne pristupaju sadržaju koji ne potiče od izvora od poverenja, što nažalost, često nije slučaj. Programeri koji kreiraju pomenute aplikacije, treba da budu pažljivi pri odabiru načina prikaza pojedinih tipova fajlova u internet pretraživačima. HTML fajlove nipošto ne treba prikazivati na način koji omogućava izvršavanje potencijalnih skripti unutar fajlova, već ih treba prikazati kao običan tekst.

5. ZAKLJUČAK

Stegosploit je vrlo efektna tehnika prikrivanja programskog koda unutar digitalnih slika formata JPEG i PNG. U ovom radu je prikazana tehnika ubacivanja neželjenog programskog koda u poliglotske JPEG i PNG fajlove. Na kraju rada su diskutovane mere bezbednosti koje omogućavaju zaštitu od potencijalnih napada kroz stegosploit.

6. LITERATURA

[1] Stegosploit, https://www.youtube.com/watch?v=np0mPy-EHII (datum poslednjeg pristupa 14.09. 2019)

[2] Dr. M. Umamaheswari, Prof. S. Sivasubramanian , S. Pandiarajan *"Applied Optimal Control"*, 2010.

[3] https://stegosploit.info/ (datum poslednjeg pristupa 14.09. 2019)

[4] Data compression, https://en.wikipedia.org/wiki/Data_compression (datum poslednjeg pristupa 14.09. 2019)

[5] JPEG format podataka, https://en.wikipedia.org/wiki/JPEG_File_Interchange_Format (datum poslednjeg pristupa 14.09. 2019)

[6] PNG format podataka, https://www.w3.org/TR/2003/REC-PNG-20031110/ (datum poslednjeg pristupa 14.09. 2019)

Kratka biografija:

Aleksandra Mišić rođena je 1994. godine u Novom Sadu. Završila je Gimnaziju "Jovan Jovanović Zmaj" u Novom Sadu, 2013. godine. 2015. godine postala je stipendista u firmi "Schneider Electric DMS NS", a osnovne akademske studije završila je 2018. godine na Fakultetu tehničkih nauka u Novom Sadu.