TESTIRANJE NA PROBOJ U FUNKCIJI PROAKTIVNE FORENZIKE I ZAŠTITE INFORMACIJA

PENETRATION TESTING IN THE FUNCTION OF PROACTIVE FORENSIC AND INFORMATION SECURITY

Gojko Grubor, Aleksandra Pešić
Univerzitet Singidunum, Danijelova 32, Beograd
ggrubor@singidunum.ac.rs

Sažetak

U radu je opisano otkrivanje ranjivosti, analizom funkcija, strukture i operacija u elektronskom uređaju, softveru ili informacionog sistemu. U zavisnosti od vrste i klasifikacije otkrivene ranjivosti, koriste se posebne strategije koje omogućavaju izvršenje proizvoljnog koda ili naredbe na ciljni sistem. Jedan od najnaprednijih i najkorišćenijih exploit alata, za testiranje ranjivosti sistema, je *Metasploit Framework*, čije su mogućnosti i funkcionalnosti za proaktivnu forenziku i zaštitu informacija detaljnije opisane u radu.

Ključne reči: *testiranje na proboj, ranjivost, exploit, metasploilt framework, proaktivna forenzika, proaktivna zaštita*

Abstract

In this paper discovering of the vulnerabilities by analyzing the functions of the structure and the operations in the electronic device and software, or in the information system is described. With regard to the types and the classifications of the discovered vulnerability, there are special strategies that are used which enable the execution of the arbitrary code or the command to the targeted system. One of the most advanced and most used exploit tools for testing the vulnerabilities of the systems is *Metasploit Framework* whose possibilities and functionalities for proactive digital forensic and information security are explained in detail in this paper.

Key words: *penetration testing, vulnarability, exploit, metasploilt framework, proactive forensic, proactive information security*

1. UVOD

Testiranje sistema zaštite informacija na proboj (*Penetration testing*) ima ključnu ulogu u eksploatisanju ranjivosti, otkrivene u okruženju ciljne mreže. Otkrivene ranjivosti mogu se koristiti za defanzivne i ofanzivne namene. U *prvom slučaju* otkrivene ranjivosti omogućavaju preventivnu zaštitu sistema – otklanjanjem ranjivosti pre pojave malicioznog *exploita* koji tu ranjivost može iskoristiti za napad na sistem, kao i proaktivnu digitalnu forenziku. U scenariju proaktivne forenzike otkrivene ranjivosti u mrežnom okruženju otklanjaju se implementacijom mehanizama zaštite (IDS/IPS[18], skenera saobraćaja, *firewalls,* log datoteka bezbednosno relevantnih događaja mrežnih uređaja itd.) koji skupljaju tragove *forenzički relevantnih bezbednosnih događaja*, odnosno, događaja koji mogu izazvati incident sa velikim posledicama, što uključuje zahteve za nadoknadu štete i sudski spor. U *drugom slučaju* otkrivena ranjivost može omogućiti etički haking za udaljeni pristup računaru u radu, u slučaju *aktivne* („*live*“)

[18] IDS/IPS (*Intrusion and Detection Systems/Intrusion and Protection Systems*) – sistemi za detekciju i sprečavanje upada u računarske sisteme i mreže.

digitalne forenzičke istrage, kada vlasnik sistema ne dozvoljava pristup ispitivanom računaru ili ne sme da zna za taj pristup. Da bi se stimulisale i istražile najbolje dostupne opcije za iskorišćavanje (eksploataciju) ranjivosti ciljnog računara, potrebno je da se sprovede pažljiva priprema, istraga i ispitivanje opcija alata i tehnika i da se primenjuju napredni alati i tehnike. Proces eksploatacije predstavlja finalnu operaciju testiranja sistema na proboj. Međutim, u slučaju zahteva za prodor u dubinu (*pivoting*) mreže, obradu i eskalaciju administratorskih prvilegija, za potrebe aktivne forenzičke istrage, rezultati mogu biti izazovni i neizvesni, zato što se tragovi prisustva moraju sakriti.

Otkrivanje ranjivosti je od ključnog značaja da bi se razumele, ispitale i testirale potencijalne ranjivosti pre napada malicioznog *exploita* (kôda, tehnike). Zatim je potrebno poznavati više dostupnih *exploit* repozitorijuma da bi se dobile informacije o javno dostupnim *exploitima* i njihovoj primeni i otklonile ranjivosti softvera bezbednosmnim popravkama (*patches*), pre malicioznog napada. Pisanje *ekploit* kôda od početka je dugotrajan i skup proces, čak i za profesionalne *penetration tester-e* gde je vreme ograničavajući faktor. Najčešće se koriste javno dostupni *exploit-i* i prilagođavaju za određeno ciljno okruženje, što može da zahteva malo više vremena i truda.

Takođe, neophodno je poznavati alate i tehnike za testiranje sistema na proboji, pa je u radu detaljnije opisana primena *Metasploit* alata, jednog od najuspešnijih za otkrivanje ranjivosti i (etički) proboj sistema.

2. ISTRAŽIVANJE RANJIVOSTI I TESTIRANJE SISTEMA NA PROBOJ

U pristupu procesu testiranja sistema na proboj, najpre treba definisati oblast i granice koje obuhvataju istraživanje ranjivosti sistema. U sistemu zaštite, ranjivosti mogu biti u *softveru, hardveru, konfiguraciji i ljudima*. U ovom radu težište je na otkrivanju ranjivosti softverskih proizvoda. Razumevanje funkcionalnosti specifičnih softverskih ili hardverskih proizvoda, može da obezbedi polaznu tačku za istragu potencijalne ranjivosti u tom proizvodu. Istraživanje ranjivosti nije lak posao, pa se zahtevaju sledeća znanja za analizu bezbednosti sistema:

- *Poznavanje programiranja* je osnova za delovanje etičkog hakera.
- *Reversni inženjering* za otkrivanje ranjivosti analizom funkcija, strukture i operacija u elektronskom uređaju, softveru ili sistemu.
- *Analiza stanja bezbednosti* za procenu uslova u kojim se ranjivost može iskoristiti, uz poznavanje izvornog kôda, pomoću automatizovanih alata ili samostalnim istraživanjem.
- *Poznavanje alata debuggers*, *data extractors*, *fuzzers*, *profilers*, *code coverage*, *flow analyzers* i *memory monitors,* koji igraju važnu ulogu u procesu otkrivanja ranjivosti i obezbeđuju konzistentno okruženje za testiranje.
- *Generisanje exploit*-a i *payload-a* čine poslednji korak u pisanju *proof-of-concept* (*PoC*) kôda za otkrivenu ranjivost. Ovo omogićava *penetration tester-u* prilagođavanje naredbi u postojećem *exploit*-u za eksploatisanje ranjivosti i pristup ciljnom računaru [2].

2.2 Napredni eksploatacioni komplet alata

Operativni sistem *BackTrack,* jedan od najpoznatijih višenamenskih alata za potrebe testiranja sistema na proboj, dolazi sa integrisanom bazom podataka *exploit-a* iz "*Offensive Security*". *Metasploit Framework - MSF* (*http://www.metasploit.com*) je jedan od alata, razvijenih u *Metasploit LLC* kompaniji. Prvobitna verzija, napravljena 2003., pisana je u *Perl* programskom jeziku, ali je kasnije, od početka do kraja, ponovo napisana u *Ruby*-ju.

Metasploit ima za cilj da obezbedi razvoj *okvira exploit*-a za *Penetration Testers-e*, kao i da pojednostavi proces razvoja *exploit-a*. *Exploit*-i su delovi kôda ili tehnike koje omogućavaju iskorišćavanje nedostataka (ranjivosti) programa i obavljanje neovlašćenih aktivnosti. Proces istraživanja ranjivosti obično sadrži sledeće faze:

Otkrivanja > *Prijava proizvođaču* > *Analiza* >Razvoj e*xploit -a*> *Testiranje* > *Objavljivanje*.

Metasploit služi i kao platforma za razvoj *payloads-a* (izvršavanje kôda nakon uspešnog pokretanja *exploit-a*), *payload encoders* (za šifrovanje *payload*-a, što podatke čine nejasnim, tako da *Intrusion Detection Systems* [*IDS*] i *Intrusion Protecion Systems* [*IPS*] ne prepoznaju i blokiraju *exploit*), ali sadrži i različite druge alate. *Metasploit,* takođe, omogućava izbegavanje antiforenzičkih alata (*Forensic Avoidance tools*), kao i niz drugih IDS tehnika.

Arhitektura *Framework-a* je podeljena na tri velike kategorije - biblioteke, interfejse i module. Interfejs (*Console, CLI, Web, GUI*) u osnovi obezbeđuje početak operativne aktivnosti kada je u pitanju bilo koja vrsta modula (*Exploits, Payloads, Auxiliaries, Encoders, Nops*). Svaki od ovih modula ima svoju specifičnu funkciju za proces testiranja:

- *Exploit* je *proof-of-concept* kôda razvijen sa ciljem da iskoristi određene ranjivosti ciljnog sistema. Telo ili struktura *exploit-a* mogu se podeliti na različite komponente, (kao što je opisano na slici 1) [4].
- *Payload* je zlonamerni kôd koji je samostalan ili je deo *exploit-a,* namenjen za pokretanje proizvoljne naredbe na sistemu.
- *Auxiliaries* je set alata namenjen za skeniranje, *sniffing (prisluškivanje)*, *wardialing*, uzimanje elektronskog potpisa, kao i druge zadatke za procenu bezbednosti.
- *Encoders* obezbeđuju izbegavanje detekcije virusa, *firewall-a*, *IDS* / *IPS* i drugih sličnih mehanizama zaštite od zlonamernih programa, kodiranjem *payload-a* u toku operacije testiranja sistema na proboj.
- *NOP* (*No Operation* ili *No Operation Performed*) je skup jezičkih instrukcija koje se često dodaju *shellcode-u*, čija je jedina funkcija da dosledno pokrije *payload* prostor [3].

2.2.1 MSFConsole

MSFConsole je centralni deo interfejsa za *penetration testers-e,* pomoću koga se mogu na najbolji način iskoristiti opcije *Framework-a* [2].

Moćno svojstvo *MSF-a* da pojednostavljuje *post-exploit* proces je *Meterpreter* modul, koji se direktno ubrizgava u pokrenuti *exploit* proces na sistemu, kao pomoćni alat za izbegavanje *IDS-a* i detekcije od strane korisnika. U testu na proboj, fokus je na prikupljanje i eksploataciju informacija, a manje na fazu nakon eksploatacije. U toj fazi se čini najveća šteta, a u njoj *Meterpreter* postaje prilično koristan. *Meterpreter* pokušava da izbegne *HIDS* (*Host Intrusion Detection Systems*), ubrizgavanjem svog kôda u već pokrenuti proces i omogućavanjem napadaču nova kodiranja i pokretanja skripti, što korumpira platformu za dalje napade. *MSF* ima podršku za bazu podataka, tako da može da stupa u interakciju sa različitim bazama podataka, kao što su *Postgres* ili *SQLite* [3].

2.2.2 MSFCLI

Interfejs komandne linije (*CLI*), kao i *MSFConsole-a,* ima podršku za različite module koji se mogu pokrenuti u brojnim slučajevima. Međutim, u poređenju sa *MSFConsole-om*, ovde nedostaju neke od naprednih mogućnosti automatizacije [2].

2.2.3 Metasploit Web Interface (MSFWeb)

MSFWeb startuje *Metasploit Web* server na 127.0.0.1 port 55555. Pretragu na ovom portu pruža uredan *Web* interfejs za *MSF*, koji će se, verovatno, više koristiti u budućnosti, jer kroz ovaj interfejs može doslovno da se "klikne i hakuje", koristeći *Metasploit*. Međutim, treba izbegavati korišćenje *Msfweb* u toku testiranja sistema na proboj, pošto se dodaje sloj aplikacije između *shell-a* i *penetration tester*-a.

2.3.1 *SNMP community* skener

Ovaj modul obavlja *SNMP sweeps* datog opsega mrežnih adresa, pri čemu se koristi poznati skup stringova i potpisa, pomoću kojih se otkriva *SNMP* informacija uređaja na ekranu [2].

2.3.2 Autentifikacioni skener VNC servera na slepo

Ovaj modul skenira opseg *IP* adresa za *VNC* (*Virtual Network Computing*) servere koji su dostupni bez zahteva za autentikaciju. Ovaj vektor napada može da postane ozbiljna pretnja za administratore sistema, jer se napadač sa interneta trivijalno može pozvati na *VNC* server [2].

2.4.1 Shellcode

To je *payload* koji treba da se izvrši nakon eksploatacije (iskorišćenja ranjivosti). U većini slučajeva se preusmerava putanja izvršenja, tako što se izvršava ubrizgavanje *payload-a*. Oni mogu da izvršavaju mnoge kompleksne operacije od otvaranja *listening socket-a*, pa do opterećenja kompajlera na udaljenom računaru [4].

2.4.2 *Bind shell*

Bind shell je udaljena *shell* konekcija koja obezbeđuje pristup ciljnom sistemu nakon uspešne eksploatacije i izvršenja *shellcode*-a, uspostavljanjem *bind port listener-a*. Ovo otvara *gateway* za povratno povezivanje napadača na ciljni računar preko *bind shell port-a,* korišćenjem alata, kao što je *netcat* koji predstavlja tunel, odnosno standardni ulaz i izlaz preko *TCP* konekcije. Ovaj primer funkcioniše na sličan način na koji *telnet* klijent uspostavlja vezu sa *telnet* serverom i okruženje u kome je napadač iza *NAT-a* (*Network Address Transformer*) ili *Firewall*-a i gde direktan kontakt sa kompromitovanog hosta ka *IP* napadača nije moguć.

2.4.3 Reverzni shell

Reverzni shell je sasvim suprotan *bind shell-u.* Umesto obavezujućeg porta na ciljnom sistemu i čekanja veze sa računara napadača, ovde se jednostavno povezuje na *IP* i port napadača, prilikom čega se stvara *shell* [2].

2.5 *Meterpreter*

Meterpreter (*Meta-Interpreter*) je napredan, nečujan, višestruki i dinamički proširiv *payload* koji deluje ubrizgavanjem *reflective DLL* u ciljnu memoriju. Skripte i dodaci mogu biti dinamički učitani za vreme izvršavanja, radi proširenja nakon aktivnosti eksploatacije. Ovo uključuje eskalaciju privilegija, kupljenje sistemskih naloga (*dumping system accounts*), očitavanje otkucaja tastature (*keylogging*), neprekidno održavanje zadnjih vrata (*persistent backdoor service*), omogućavanje udaljenog pristupa (*enabling remote desktop*) i mnoge druge maliciozne operacije. Štaviše, cela komunikacija *meterpreter shell-a* je podrazumevano šifrovana.

Pored toga, *Meterpreter* omogućava programerima da napišu svoje dodatke u obliku *DLL* datoteke koje se mogu *uploadovati* i izvršavati na udaljenom sistemu. Ali prava vrednost *Meterpreter-a* je u tome što radi na principu samoubrizgavanja na ranjive procese na udaljenom sistemu, pri eksploatacji. Sve komande koje prolaze kroz *Meterpreter* takođe se izvršavaju u kontekstu tekućih procesa. Na ovaj način, on je u stanju da izbegne otkrivanje od strane

antivirusnih programa ili osnovne forenzičke istrage. Digitalni forenzičar bi trebao da sprovede analizu memorije pokrenutih procesa.

3. ZAKLJUČAK

Iako je nemoguće napraviti potpuno bezbedan i zaštićen sistem, zahvaljujući brzom napretku IT-a i tehnologija zaštite, sve uspešnije se rešava veliki problem ekspoatisanja ranjivosti sistema u svim oblastima računarstva. U ovom radu je objašnjeno nekoliko ključnih aspekata primene alata i tehnika za testiranje ranjivosti sistema na proboj, sa dva glavna cilja - proaktivne zaštite informacija i etičkog hakinga u funkciji aktivne digirtalne forenzičke istrage. Pregled metoda istraživanja ranjivosti ističe potrebu da proces testiranja sistema na proboj zahteva brojne alate i tehnike, kao i neophodna znanja i veštine za procenu ranjivosti. Od velike koristi su *online* repozitorijumi u kojima se nalazi veliki broj javno dostupnih ranjivosti i *exploit* kôdova. Prikazana je i praktična primena naprednih *exploit* alata pod nazivom "*Metasploit Framework (MSF)*", koji su dizajnirani za istragu, određivanje cilja i eksploataciju realizovanog napada. Dat je i uvid u razvoj *exploit-a,* kroz analizu svakog koraka *exploit* kôda iz *MSF-a* koji omogućava razumevanje osnovne strukture i izgradnje strategije napada i eksploatacije [5].

MSF, moćna i fleksibilna *exploit* razvojna platforma, dospeo je u fazu u kojoj može da izvrši testiranja stanja bezbedonosti računarskih i mrežnih sistema. Prerađen u *Ruby* programskom jeziku, poseduje mogućnost proširenja i promene implementacije izlaza iz drugih alata, kao što su *Nmap* i *Nessus*. Takođe, nudi veliki broj interfejsa - popularna *msfconsole* sada je proširena sa istovremenim sesijama i izvršenjima *exploit-a*, *msfweb* za *Web* interakcije, *msfcli* za komandnu liniju izvršenja *exploit-a* i msfd za eksploataciju u *daemon* modu. Baze podataka *msfencode* i *msfopcode* dozvoljavaju korišćenje *ekploit-a,* tako da odgovara okruženju ciljnog računara.

Metasploit je više od *ekploit* repozitorijuma. Kao kompletna platforma za istraživanje bezbednosti sistema i proaktivnu zaštitu, kao i aktivnu digitalnu forenzičku istragu u funkciji etičkog hakinga, alat omogućava krajnim korisnicima da prilagode *framework* svojim potrebama. Iako *exploit payload* poseduje dovoljno funkcionalnosti za većinu korisnika, sposobnost da se *payload* proširi predstavlja novi kvalitet u primeni. Relevantan primer ovoga je nova funkcija - *hooking HIPS*. Tipično, *hips* koji se oslanja na funkciju *hooking* pronalazi funkcije od značaja za napadače i zamenjuje njihov početak ili uvod, sa eksplicitnim *jump* na izvršavanje glavne analize.

Metasploit automatizacija je mnogo više od pokretanja *exploit-a* kroz širok izbor mreža i ciljnih računara. Zapravo se radi o automatizaciji onog što se dešava nakon uspešnog iskorišćenja ranjivosti (eksploatacije). S obzirom da skeneri ranjivosti u stvari, ne preuzimaju kontrolu nad *hostom*, nije moguće bilo šta uraditi posle eksploatacije, kao što je dodavanje korisnika (iz bezbednosnih razloga) ili čak preuzimanje i instaliranje bezbednosnih popravki (*patches*) na ranjivost koja, na prvom mestu, dozvoljava kontrolu nad *hostom*.

Očigledno u primeni *MSF*-a u funkciji etičkog hakinga u sprezi sa aktivnom forenzičkom istragom, veliku ulogu igra socijalni inženjering posebno u situaciji kada vlasnik sistema ne dozvoljava fizički pristup sistemu u radu, ili kada postoji samo osnovana sumnja pa se aktivnosti forenzičke istrage moraju sakriti od vlasnika sistemane.

Proaktivna zaštita informacija i proaktivna forenzika testiranjem sistema na proboj najbolje su odbrane, jer zahtevaju redovno ažuriranje sistemskog softver sa bezbednosnim popravkama, jačanje sistema zaštite novim mehanizmima, kao i bezbedonosnu obuku zaposlenih za otklanjanje otkrivenih ranjivosti konfiguracija, hardvera i ljudskog faktora. Na taj način se u

velikoj meri ublažava rizik od napada i zlonamernog iskorišćavanja ranjivosti sistema. Pored toga, dublji prodor u operativne sisteme i aplikacije, može unaprediti pisanje boljih, sigurnijih kôdova, što je od neprocenjive vrednosti za bezbednost informacija i poslovanje organizacija [5].

LITERATURA

[1] Maynor, D., Mookhey, K. K., *Metasploit Toolkit for Penetration Testing, Exploit Development, & Vulnerability Research*, Syngress Publishing, Inc. 2007, Burlington, MA.

[2] Shakeel, A., Heriyanto, T., *BackTrack 4: Assuring Security by Penetration Testing*, First published, Packt Publishing 2011, Birmingham, UK

[3] *Metasploit Alternate Uses for a Penetration Test*, http://greyhat-security.com/ (posećeno: 16.01.2012.)

[4] http://www.symantec.com/connect/articles/metasploit-framework-part-1, (posećeno: 14.01.2012).

[5] Pešić Aleksandra, *"Testiranje sistema na proboj u cilju zaštite informacija''*, Univerzitet Singidunum, 2011.

[6] Ivan Bütler, *Using Nmap results in Metasploit with db_autopwn,* (posećeno: 01.06.2011).