Penetraciono testiranje

Izveštaj testiranja sprovedenog nad aplikacijom Payment Service Provider

1. Osnovni pojmovi

Penetraciono testiranje predstavlja postupak specifičnog testiranja sistema, pri kom se tester ponaša kao napadač, a sve sa ciljem pronalaska ranjivosti koje se u produkcionim okruženju mogu iskoristiti i formirati se vektor napada takav da on rezultuje uspešnom eksploatacijom sistema. Upravo ovakvim pristupom testiranju pokušavaju se predupredi zlonamerni korisnici tako što će ranjivosti koje sistem u određenom trenutnku poseduje biti pravovremeno otkrivene i otklonjene.

Penetracionom testiranju može podleći bilo koji deo sistema, u zavisnosti od njegove prirode i okruženja u kom se nalazi. Ukoliko se kao primer uzme veb aplikacija, penetraciono testiranje može se vršiti nad aplikacijom, aplikativnim serverom, operativnim sistemom fizičkog servera, mrežnim interfejsima servera, mrežnom infrastrukturom od koje taj server zavisi, bazom podataka, serverom baze podataka itd.

Kako napadač sistema može biti subjekt koji ne poseduje nikakvo znanje o sistemu (sem npr. javne IP adrese preko koje se može pristupiti funkcionalnostima sistema), neko ko ima detaljne informacije o infrastrukturi sistema, source code-u aplikacije, korišćenim tehnologijama itd. ili bilo ko na spektru između dve navedene krajnosti, razlikuju se pristupi penetracionom testiranju tako da se svaki može mapirati na određenu grupu navedenih napadača. U odnosu na nivo detaljnosti informacija koje su inicijalno pružene testeru, sistem za njega može predstavljati black, white ili gray box.

Testovi koji se izvode nad sistemom mogu biti manuelni ili automatski. Manuelne testove direktno sprovodi osoba, tačnije tester, dok se automatski testovi vrše upotrebom nekog od alata za penetraciono testiranje. Dok su alati koji automatski vrše testove pogodni za sprovođenje velike količine testova u jedinici vrmeena i efikasni u pronalaženju osnovnih ranjivosti sistema, manuelnim testovima se osigurava da će biti istestirani i delovi sistema kojima alat nije mogao da pristupi ili nije namenjen za njih, kao i da će biti formirani specifični testovi za dati sistem, što alat koji je namenjen da testira širok spektar sistema ne poseduje sam po sebi.

2. Proces i rezultati testiranja

Prvi korak testiranja bio je upotreba alata OWASP ZAP, open source i besplatnog alata namenjenog za penetraciono testiranje. Alat podržava veliki skup funkcionalnosti koji podrazumeva aktivno i pasivno skeniranje aplikacije, praćenje sesije, automatsku i manuelnu izmenu zahteva koji se šalju kako bi se targetirale specifične ranjivosti sistema itd. Pasivno skeniranje postiže se ručnim istraživanjem aplikacije ili spideringom (crawling sajta koji za cilj ima prikupljanje informacija o strukturi i sadržaju stranica od kojih se aplikacija sastoji). Prilikom pasivnog skeniranja, alat samo prati zahteve i odgovore koji se razmenjuju, ne vršeći nikakvu izmenu, sa ciljem detektovanja mogućih ranjivosti. Za razliku od toga, aktivno skeniranje se vrši napadanjem sistema, odnosno izmenom zahteva koji se šalju, a oni

se formiraju tako da bi eksploatisali sistem koji testiraju. Dok se vrši testiranje, ranjivosti se detektuju i testeru prikazuju kroz *alerts* tab i mogu imati različit stepen rizika (*high, medium, low, informational*).

Proces testiranja započet je ručnim istraživanjem svih stranica koje aplikacija poseduje i pozivanjem svih funkcionalnosti koje su ponuđene. Nakon toga izvršeno je automatsko skeniranje stranica upotrebom AJAX spider-a kako bi informacije bile potpunije i kako bi se pronašli delovi aplikacije koji su eventualno ručno preskočeni. Procesom pasivnog skeniranja otkriveno je da postoje aplikacije koje slušaju na portovima 4201 i 8090 koje bi trebalo dalje testirati aktivnim skeniranjem, što je i bio naredni korak testiranja. U ovom koraku, šalju se zahtevi automatski kreirani od alata koji pokušavaju pronaći različite ranjivosti. Slika 1 prikazuje ranjivosti na koje alat cilja prilikom vršenja aktivnog skeniranja.

	Strength
Analyser	
Plugin	
Path Traversal	Medium
Remote File Inclusion	Medium
Source Code Disclosure - /WEB-INF folder	Medium
External Redirect	Medium
Server Side Include	Medium
Cross Site Scripting (Reflected)	Medium
Cross Site Scripting (Persistent) - Prime	Medium
Cross Site Scripting (Persistent) - Spider	Medium
Cross Site Scripting (Persistent)	Medium
SQL Injection	Medium
Server Side Code Injection	Medium
Remote OS Command Injection	Medium
Directory Browsing	Medium
Buffer Overflow	Medium
Format String Error	Medium
CRLF Injection	Medium
Parameter Tampering	Medium
ELMAH Information Leak	Medium
.htaccess Information Leak	Medium
Script Active Scan Rules	Medium
Cross Site Scripting (DOM Based)	Medium
SOAP Action Spoofing	Medium
SOAP XML Injection	Medium

Slika 1 - Ranjivosti koje OWASP ZAP proverava prilikom aktivnog skeniranja

Kada je automatsko aktivno skeniranje obavljeno, nad obe apliakcije urađen je forced browsing kako bi se pokušalo pristupiti fajlovima i/ili direktorijumima iz predefinisane liste sa kojom OWASP ZAP dolazi, a čijim bi se uspešnim pristupanjem pronašle ranjivosti sistema. Sistem nije podlegao ovoj vrsti napada.

Naredni pokušaj pronalaženja ranjivosti bio je kroz testiranje SQL Injection i XSS napada nad REST endpoint-ima koji prihvataju parametre koji se mogu formirati tako da dovedu do nekog od navedenih Injection napada. Kako bi testovi bili izvršeni, upotrebljena je Fuzzer funkcionalnost koju OWASP ZAP nudi i koja radi tako što se deo zahteva modifikuje definisanim vrednostima, a na slikama 2 i 3 prikazan je deo skupa vrednosti kojima se zahtev menja kada se pokušava vršiti SQL Injection odnosno XSS napad. Ni za jedan od maliciozno formiranih zahteva sistem nije bio eksploatisan.

```
  29: ' or 'unusual' = 'unusual'
  30: ' or 'something' = 'some'+'thing'
  ...

SQL Injection
  1: a
  2: a'
  3: a' --
  4: a' or 1=1; --
  5: @
  6: ?
  7: ' and 1=0) union all
  8: ? or 1=1 --
  9: x' and userid is NULL; --
  10: x' and email is NULL; --
  11: anything' or 'x'='x
  12: x' and 1=(select count(*) from tabname); --
  13: x' and members.email is NULL; --
  14: x' or full_name like '%bob%
  15: 23 or 1=1; --
  16: '; exec master..xp_cmdshell 'ping 172.10.1.255'--
```

Slika 2 - Pokušaji SQL injection napada

```
URI Cross Site Scripting
  1: aim: &c:\windows\system32\calc.exe" ini="C:\Documents and Settings\All Users\Sta
  2: firefoxurl:test|"%20-new-window%20javascript:alert(\'Cross%2520Browser%2520Scrip
  3: navigatorurl:test" -chrome "javascript:C=Components.classes;I=Components.interfa
  4: res://c:\\program%20files\\adobe\\acrobat%207.0\\acrobat\\acrobat.dll/#2/#210

URL Breaking
  1: httP://aa"><script>alert(123)</script>
  2: httP://aa'><script>alert(123)</script>
  3: httP://aa<script>alert(123)</script>

XSS 101
  1: <script>alert('xss')</script>
  2: <script>alert(string.fromcharcode(88,83,83))</script>
  3: </title><script>alert(1)</script>
  4: '> <script>alert(3)</script>
  5: `> <script>alert(5)</script>
  6: > <script>alert(4)</script>
  7: </title><script>alert(1)</script>
  8: <<script>alert("xss");//<</script>
  9: >"'
```

Slika 3 - Pokušaji XSS napada

Fuzzer funkcionalnost upotrebljena je i za pokušaj Brute-force napada prilikom prijave na sistem kako bi se otkrile lozinke registrovanih korisnika. Zahtev je modifikovan tako da se za zadati username pošalje zahtev za svaku od 10000 najgorih lozinki (lozinke preuzete sa sledeće lokacije https://github.com/OWASP/passfault/blob/master/wordlists/wordlists/10k-worst-passwords.txt). Nakon testiranja utvrđeno je da nalogu korisnika nije bilo moguće pristupiti, prvenstveno zbog toga što sistem ni ne dozvoljava registraciju korisnika sa lozinkom koja ne zadovoljava rigorozne uslove. Međutim i kada bi lozinke bile odabrane na sofisticiraniji način, napad je onemogućen tako što se nalog korisnika blokira na 24 časa nakon tri neuspešna pokušaja. Nedostatak koji postoji je odsustvo multi-factor authentication mehanizma. Ukoliko maliciozni korisnik dođe u posedstvo lozinke registrovanog korisnika, on može pristupiti tom nalogu zato što je

jedini uslov za pristup poznavanje lozinke. Dodatni nedostatak koji je primećen je predugo trajanje (3h) jwt tokena koji se dodeli korisniku koji se uspešno autentifikovao. Time se dolazi u opasnost da računaru na kom je sačuvan token može pristupiti neko drugi i imati direktan pristup nalogu.

Kako bi se testirale Broken Access Control ranjivosti, ručno se pokušalo pristupiti svim stranicama koje aplikacija poseduje sa ciljem pristupa stranicama kojima trenutni korisnik ne bi smeo imati pristup. Napadi ovog tipa bili su neuspešni. Ista stvar je isprobana nad API-jem aplikacije gde je neautentifikovani I/ili neautorizovani korisnik pokušao slati zahteve, međutim svaki od njih završavao je neuspešno. Takođe, isprobano je dobavljanje podataka o drugim korisnicima od strane korisnika koji se autentifikovao, ali i taj pokušaj bio je neuspešan. Nedostatak koji je pronađen je da postoje endpoint-i koji su sakriveni iza gateway-a i niko im kroz gateway ne može pristupiti, međutim otkrivanjem lokacije pojedinačnog mikroservisa zlonamerni korisnik imao bi pristup i toj putanji, iako je ona namenjena samo drugim mikroservisima. Kako bi ovaj problem bio rešen, preporučuje se autentifikacija u komunikaciji među mikroservisima, bilo upotrebom tokena ili mTLS protokola pri uspostavljanju komunikacije.

Kako bi se otkrile dodatne površine za napad, izvršeno je skeniranje zavisnosti koje servisi poseduju, kako bi se sistem mogao eksploatisati kroz nedostatke koje poseduju biblioteke na koje se on oslanja. Rezultati skeniranja mogu se pronaći u direktorijumu *dependencies*.

Poslednji tip testiranja izvršen je nad sistemom za upravljanje bazama podataka. Iako se kroz aplikaciju nije direktno mogao vršiti injection i time manipulisati podacima u bazi podataka, sistem za pristup bazama koristi generičke kredencijale koje maliciozni korisnik može iskoristiti ukoliko sazna adresu servera baze podataka i na taj način može pristupiti kompletnim podacima koje sistem poseduje. Iako su vršena enkripcija osetljivih podataka i heširanje lozinki, omogućen pristup podacima predstavlja površinu za napad koja može biti osnova za dalje napade napade pri kojima bi se mogla izvršiti dekripcija zaštićenih podataka ili otkrivanje lozinki korisnika.