Sigurnost Java aplikacija – 5 najčešćih grešaka

Luka Milković
Luka.Milkovic@infigo.hr

Bojan Ždrnja, CISSP, GCIA, GCIH
Bojan.Zdrnja@infigo.hr

INFIGO IS http://www.infigo.hr

Sadržaj

- Tko smo?
- Java je sigurna!
  - ... ili možda ipak nije?
- Najčešće Java ranjivosti
  - pogled na Java EE aplikacije
    - iz iskustva INFIGO IS
  - kako se zaštititi?
- Zaključak

INFIGO IS

- Najkompetentnija hrvatska tvrtka na području informacijske sigurnosti
  - višegodišnje iskustvo na području sigurnosti informacijskih sustava
  - najpoznatiji na području penetracijskog testiranja
    - sadržaj ove prezentacije proizašao je iz desetaka testiranih domaćih Java EE aplikacija
- Savjetodavne usluge, provjera sigurnosti, *managed security services* (Splunk, Sourcefire, Rapid7, Nagios)

Java je sigurna

- Svi znamo da je Java sigurna
  - *type-safe*
  - automatsko upravljanje memorijom
  - *garbage collection*
  - provjera veličine polja
  - provjera *bytecodea* (*bytecode verifier*)
    - štiti od čitavog niza potencijalnih sigurnosnih ranjivosti
      - nelegitiman *bytecode*
      - *stack overflow/underflow*
      - neispravno rukovanje podacima
- No, greške se ipak dešavaju ...

Java je sigurna

Blackhole β | СТАТИСТИКА | ПОТОКИ | ФАЙЛЫ | БЕЗОПАСНОСТЬ | НАСТРОЙКИ | Выйти

Начало: Конец: Применить Автообновление: 5 сек.

СТАТИСТИКА

ЗА ВЕСЬ ПЕРИОД — 10.32% ПРОБИВ

13289 хиты 11506 хосты 1187 загрузки

ЗА СЕГОДНЯ — 11.55% ПРОБИВ

3013 хиты 2760 хосты 300 загрузки

ПОТОКИ	ХИТЫ ↑	ХОСТЫ	ЗАГРУЗКИ	%
DENIS	13285	11505	1187	10.32
default	4	3	1	0.00

БРАУЗЕРЫ	ХИТЫ	ХОСТЫ	ЗАГРУЗКИ	% ↑
Chrome	2273	2148	485	22.58
Mozilla	104	72	11	15.71
Firefox	5033	4847	581	11.99
Opera	360	288	22	7.75
MSIE	4232	3080	77	2.51
Safari	1287	1102	11	1.00

ОС	ХИТЫ	ХОСТЫ	ЗАГРУЗКИ	% ↑
Windows 2003	21	18	5	27.78
Windows 2000	41	22	4	18.18
Linux	179	143	19	13.48
Windows XP	3838	3206	399	12.48
Windows 7	5059	4490	478	10.66
Windows Vista	3173	2752	264	9.61
Mac OS	978	900	18	2.00

ЭКСПЛОИТЫ	ЗАГРУЗКИ	% ↑
Java X	584	49.20
Java SMB	460	38.75
PDF	108	9.10
Java DES	29	2.44
MDAC	6	0.51

СТРАНЫ	ХИТЫ ↑	ХОСТЫ	ЗАГРУЗКИ	%
United States	12417	10981	1119	10.19
Brazil	154	101	9	8.91
India	63	35	4	11.43
Japan	47	9	3	33.33
Mexico	37	28	0	0.00
Argentina	31	12	2	16.67
Bulgaria	31	10	0	0.00
Indonesia	29	17	5	29.41
Romania	26	16	0	0.00
Pakistan	26	13	1	7.69
Philippines	24	16	1	6.25
Israel	22	14	2	14.29
Chile	19	6	0	0.00
Singapore	18	15	0	0.00
Hungary	18	11	0	0.00
Другое	327	222	41	18.55

Создать виджет

Blackhole v.1.0.0

Java je sigurna

ЭКСПЛОИТЫ	ЗАГРУЗКИ	% ↑
Java X	584	49.20
Java SMB	460	38.75
PDF	108	9.10
Java DES	29	2.44
MDAC	6	0.51

- Ova prezentacija koncentrirati će se na poslužiteljsku stranu

Java EE aplikacije

- Jedno od najčešćih okruženja za razvoj poslovnih aplikacija
  - srećemo vrlo često u penetracijskim testovima
- Veliki broj internih aplikacija
  - malo pažnje obraća se sigurnosti
    - možda uzrokovano i neinformiranošću programera
      - „Java je sigurna”
- Ranjivosti Java EE aplikacija mapirane prema OWASP Top 10 ranjivostima
  - http://www.owasp.org
  - Open Web Application Security Project
    - cilj povećati sigurnost web aplikacija

Najčešće ranjivosti Java aplikacija

- A1 – Injection
- A2 – Cross Site Scripting (XSS)
- A3 – Broken authentication and Session Management
- A4 – Insecure Direct Object References
- A5 – Cross Site Request Forgery (CSRF)

A1 - Injection

- Ranjivosti neispravnog rukovanja podacima primljenim od korisnika
  - nastaju kada ih aplikacija dalje predaje interpreteru
  - ili koristi kao dio naredbe ili upita
- Najpoznatije SQL Injection ranjivosti
  - nastaju kada se primljeni podaci koriste za formiranje SQL upita
  - još uvijek vrlo česte!
- Postoji još čitav niz Injection ranjivosti
  - umetanje naredbi operacijskog sustava
  - LDAP Injection
  - XPATH Injection

A1 - Injection

- Injection ranjivosti se obično relativno jednostavno iskorištavaju
  - ujedno ih je i u većini slučajeva jednostavno otkriti
- Primjer ranjivosti:

```
String upit = "SELECT iznos FROM racun WHERE broj_racuna = "
    + request.getParameter("broj_racuna");

try {
    Statement statement = connection.createStatement( ... );
    ResultSet rezultat = statement.executeQuery( upit );
    }
```

A1 - Injection

- Kako se zaštititi?
  - korištenje pripremljenih upita (*prepared statement*)
    - ovo je preporučen način zaštite
  - ispravno rukovanje kritičnim znakovima (*escaping*)
    - potrebna je velika pažnja – metode napada svakim su danom sve bolje, baze se mijenjaju itd ...

```
String broj_racuna = request.getParameter("broj_racuna");
String upit = "SELECT iznos FROM racun WHERE broj_racuna = ? ";

PreparedStatement psupit = connection.prepareStatement( upit );
psupit.setString( 1, broj_racuna );
ResultSet rezultat = psupit.executeQuery();
```

A2 – Cross Site Scripting (XSS)

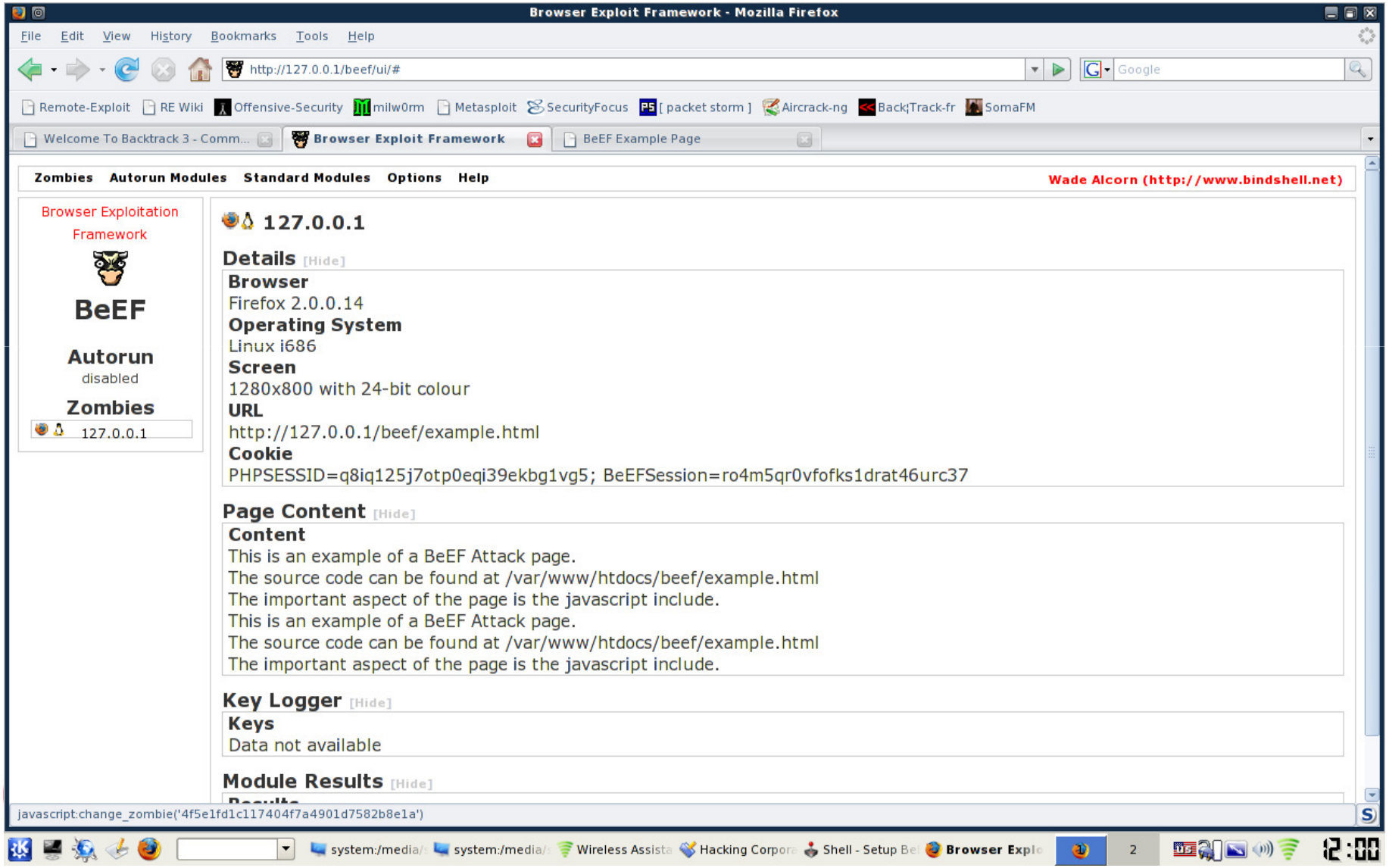
- Slučaj kada aplikacija prikazuje podatke bez prethodne provjere/enkodiranja
- Omogućava napadačima izvođenje proizvoljnih skripti u žrtvinom web pregledniku
  - JavaScript, VBScript ili jednostavno modificiranje prikazanog HTML-a
- Tri vrste XSS ranjivosti:
  - reflektirane
  - pohranjene
  - DOM

A2 – Cross Site Scripting (XSS)

```
String rezultat = "<input name='kartica' type='TEXT' value='"
    + request.getParameter("CC") + "'>";
```

- Napadač jednostavno zatvara tag i unosi svoj kôd
- XSS ranjivosti se vrlo jednostavno otkrivaju i iskorištavaju
  - postoji čitav niz metoda za iskorištavanje XSS ranjivosti
    - pa čak i gotovih paketa poput BeEF (Browser Exploitation Framework)
      - čak integriran s Metasploitom

BeEF


Browser Exploit Framework - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1/beef/ui/#
Remote-Exploit RE Wiki Offensive-Security milw0rm Metasploit SecurityFocus [packet storm] Aircrack-ng BackTrack-fr SomaFM
Welcome To Backtrack 3 - Comm...
Browser Exploit Framework
BeEF Example Page
Zombies Autorun Modules Standard Modules Options Help
Wade Alcorn (http://www.bindshell.net)
Browser Exploitation Framework
BeEF
Autorun
disabled
Zombies
127.0.0.1
127.0.0.1
Details [Hide]
Browser
Firefox 2.0.0.14
Operating System
Linux i686
Screen
1280x800 with 24-bit colour
URL
http://127.0.0.1/beef/example.html
Cookie
PHPSESSID=q8iq125j7otp0eqi39ekbg1vg5; BeEFSession=ro4m5qr0vfofks1drat46urc37
Page Content [Hide]
Content
This is an example of a BeEF Attack page.
The source code can be found at /var/www/htdocs/beef/example.html
The important aspect of the page is the javascript include.
This is an example of a BeEF Attack page.
The source code can be found at /var/www/htdocs/beef/example.html
The important aspect of the page is the javascript include.
Key Logger [Hide]
Keys
Data not available
Module Results [Hide]
javascript:change_zombie('4f5e1fd1c117404f7a4901d7582b8e1a')

A2 – Cross Site Scripting (XSS)

- Kako se zaštititi?
  - Svi podaci koji se prikazuju trebaju biti ispravno enkodirani
    - bez obzira na mjesto
    - postoje XSS ranjivosti koje se mogu iskoristiti samo u pojedinim web preglednicima
  - Preporučeno je provesti i filtriranje primljenih podataka
    - uvijek filtriranje prema bijeloj listi, nikad prema crnoj
  - Korištenje HttpOnly opcije otežava provođenje XSS napada
    - ali ih ne onemogućava

```
& --> &amp;
< --> &lt;
> --> &gt;
" --> &quot;
' --> &#x27;
/ --> &#x2F;
```

```
String sessionid = request.getSession().getId();
response.setHeader("SET-COOKIE", "JSESSIONID=" + sessionid + "; HttpOnly");
```

A3 – Broken Authentication and Session Management

- Ranjivosti koje nastaju kod neispravnog rukovanja sjednicama
  - neispravna zaštita podataka o sjednici ili sigurnosnih tokena
  - uključuje i neispravno brisanje sjednice
- Najčešća mjesta gdje se nalaze ove ranjivosti uključuju:
  - logout funkciju
  - ekran za upravljanje zaporkama
  - provjera isteka sjednice
  - „Remember me” funkcija
  - tajna pitanja kod zaboravljenih zaporki

A3 – Broken Authentication and Session Management

- Često se sjednički podaci nalaze u URL adresama
  - www.test.hr/stranica;jsessionid=2P0OC2JDPXM0OQSNDLPSKHCJUN2JV?korisnik=1
    - problem ovakvih URL-ova je da se vide u *proxy* sustavima i logovima ciljnog poslužitelja
- Aplikacije često kritične podatke lokalno pohranjuju na nesiguran način
  - npr. korisnička imena i zaporke
    - uvijek, obavezno koristiti *salt* + *hash*

```
import java.security.MessageDigest;

public byte[] getHash(String password, byte[] salt) throws NoSuchAlgorithmException {
    MessageDigest digest = MessageDigest.getInstance("SHA-256");
    digest.reset();
    digest.update(salt);
    return digest.digest(password.getBytes("UTF-8"));
    }
```

A3 – Broken Authentication and Session Management

- Kako se zaštititi?
  - potrebno je ispravno definirati autentikacijske procese, kao i procese upravljanja sjednicama
  - sjedničke informacije i tokeni moraju se sigurno pohranjivati
    - preporuka je u Cookie
  - nakon uspješne autentikacije uvijek napraviti novu sjednicu
  - osigurati da odjava briše sve podatke o aktivnoj sjednici
  - osjetljive podatke nikad ne postavljati u URL

A4 – Insecure Direct Object References

- Parametri koji dolaze od korisnika često se koriste za dohvaćanje objekata
  - kroz SQL ili bilo koji drugi način
- Ničemu što dolazi od korisnika ne smije se vjerovati!
  - www.test.hr/stranica.jsp?korisnik=42
  - napadač isprobava 43, 44, 45, 46 ...

```
String upit = "SELECT * FROM racuni WHERE racun = ?";
PreparedStatement pstmt = connection.prepareStatement(upit , ... );
pstmt.setString( 1, request.getParameter("racun"));
ResultSet results = pstmt.executeQuery();
```

A4 – Insecure Direct Object References

- Kako se zaštititi?
  - programski napraviti indirektne reference na objekte
    - trebaju ovisiti o prijavljenom korisniku ili sjednici
  - provjeriti pravo pristupa traženom objektu
    - napraviti za svaki objekt
- Ova kategorija ranjivosti sreće se vrlo često
  - obično su kritične

A5 – Cross Site Request Forgery (CSRF)

- CSRF ranjivosti napadaču omogućavaju provođenje aktivnosti pod privilegijama trenutno prijavljenog korisnika
  - bazira se na činjenici da se sjednica pohranjuje u *Cookie*
    - web preglednik automatski šalje Cookie prilikom svakog upita

```
<img src="http://www.test.hr/admin/deletemessage?mes_id=42" />
```

- Ranjivost se može iskoristiti i u POST upitima
  - GET je još jednostavniji

A5 – Cross Site Request Forgery (CSRF)

- Kako se zaštititi?
  - ne koristiti GET upite za osjetljive/kritične aktivnosti
    - ne sprječava CSRF, samo malo otežava provođenje napada
  - napraviti jedinstveni, slučajni token za svaku stranicu
    - prilikom POST upita token se šalje natrag aplikaciji
    - aplikacija provjerava ispravnost tokena
- OWASP CSRFGuard projekt
  - https://www.owasp.org/index.php/Category:OWASP_CSRFGuard_Project
  - Java EE filter, koristi tokene

Zaključak

- OWASP Top 10 ranjivosti web aplikacija prisutne su i u Java aplikacijama
- Danas se sve više susrećemo s logičkim ranjivostima u aplikacijama
  - obratiti pažnju na poslovnu logiku
- Korisničkom unosu ne može se vjerovati!
  - sve uvijek provjeriti na strani poslužitelja
- Kritične aplikacije testirati
  - penetracijski testovi od strane iskusnih osoba
- OWASP Enterprise Security API
  - https://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API