Security vulnerabilities in new web applications

Ing. Pavol Lupták, CISSP, CEH
Lead Security Consultant

Introduction

$whoami

Pavol Lupták

- 10+ years of practical experience in security and seeking vulnerabilities with a strong theoretical background (CISSP, CEH)
- OWASP Slovakia Chapter Leader
- co-author of the OWASP Testing Guide v3.0
- owner of security company Nethemba s.r.o. focused on penetration tests and web application security audits

The goal of this presentation

- To show security vulnerabilities common in new web applications (Internet banking, e-shops, online casinos, ..) during the period 2008-2009
- “Real data” from plenty of new web applications developed by Czech/Slovak companies was used
- To ensure their privacy, the used data was aggregated

SQL injection downtrend, risk of second-order injection vulnerabilities

- Most new applications use prepared statements/parametrized queries
- Complex architecture with multiple different applications can increase the likelihood of second-order injection vulnerabilities
- New fully-automatized SQL injection tools are available (sqlmap, Power SQL injector)

Increased complexity of XSS vulnerabilities, AJAX security

- Due to a huge number of XSS variations, "**blacklisting**" simply does not work
- Importance of the "**output validation**" (inevitable if the stored data is contaminated with injected code)
- It is still a problem to automatically reveal persistent XSS / AJAX vulnerabilities
- New XSS demo tools (Browser Rider, Beef)

Vulnerabilities in session management

- Many new applications use own session management that is almost always bad implemented
- If it is possible, use language underlying session management instead
- At this moment, there is no automated way to fully test session management security

Session Fixation Attacks

- If the attacker can inject an arbitrary value in anonymous session tokens and the application accepts this value and uses it for authenticated login, session fixation attacks are feasible
- Can be used to hijack user sessions using social engineering, XSS vulnerabilities, ..
- The application should not accept injected session token, should regenerate it after login, invalidate after logout

Absence of “secure” and “HttpOnly” flag for cookies

- Secure flag prevents the browser to send a cookie through unencrypted connection
- HttpOnly flag prevents the injected javascript code to steal cookies using the document.cookie parameter
- HttpOnly is effective only when TRACE/TRACK HTTP methods are disabled on web server

Simultaneous sessions

- In normal circumstances there is no need for simultaneous sessions for one user in the application
- User should be informed if another user with the same credentials is logged to the application
- Bind the user session with its IP address/subnet
- Session logging is very important

Brute force attack against session management

- Feasible when session token is shorter than 128 bits or it is easily determinable
- Almost no application can detect this attack because session token is not associated with an existing user
- The application should detect increased number of session tokens from one IP address in a short time and blocks it for a defined period

CSRF vulnerabilities

- GET requests are still used for sensitive operations
- Many ways how to protect against CSRF:
- Requiring special non-determinable parameters in GET/POST requests (hidden fields)
- AJAX “double submit” cookies
- Using other verification channels (email, SMS, ..)

Weak and broken CAPTCHA

- We successfully break all our tested CAPTCHAs with CAPTCHA killer
- It is really difficult to create strong CAPTCHA
- Many CAPTCHA implementations are vulnerable to replay attacks
- Is still using CAPTCHA a good idea?

Business Logic Security Flaws

- Still prevailed because of complexity to detect them automatically
- Common problems:
- Permanent user's account locking
- Using negative numbers to gain a lot of money
- Enumeration of users using “forgotten password” or “registration” form

Vulnerable SSL protocol, weak ciphers, hashes and passwords

- SSLv2 is still massively used
- Weak (less than 56-bits) SSL ciphers are used
- MD5 is still massively used
- DES ECB is still used (in a bank environment!)
- Applications have password complexity restrictions that drastically decrease all possible combinations
- Short complicated passwords are still used
- Hashes are not salted (risk of rainbow table attacks)

Any questions?

Thank you for listening

Ing. Pavol Lupták, CISSP CEH