How to Prevent Business Flaws Vulnerabilities In Web Applications

Marco Morana
OWASP

OWASP
Cincinnati Chapter,
January 2011 Meeting

Copyright © 2010 - The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License.

The OWASP Foundation
http://www.owasp.org
What is OWASP?
Agenda For Today’s Presentation

1. General Background on Business Logic Attacks (BLA)
   1. Problem statement
   2. Business logic exploits
   3. Categorization of business logic flaws

2. Vulnerability Analysis of Business Logic Flaws
   1. Threat, vulnerabilities, and attacks
   2. Root causes of vulnerabilities leading to BLA
   3. Categorization of business logic flaws using OWASP T10, SANS-25, WASC

3. Identification and mitigation of Business Logic Flaws
   1. Application Threat Modeling
   2. Testing for Business Logic Flaws
   3. Countermeasures

4. Q & A
Business Logic Attacks
Business Logic Attacks: The Problem Statement

- Target specifically the business rules of the application by abusing them for monetary gain and fraud, some examples:
  - Attacking shopping cart transactions to alter price of an item before checkout
  - Attack shopping cart after checkout to bypass credit card purchase validation before shipping the item
  - Attacking money transfers in an on-line banking transactions by exploiting weaknesses in account validations

- Take advantage of overlooked flaws in enforcing strict control of business logic at the application layers, for example:
  - Security design reviews does not focus on designing controls for preventing abuses of business logic
  - Automated vulnerability assessments and code scan do not found business logic flaws
How to Game CNBC's Stocks Contest

Contestants in a CNBC stock-picking game may have been able to game the cable channel's trading system in the quest for a $1 million prize

By Tim Catts

Cable business news channel CNBC attracted some 375,000 players to its recent “Million Dollar Portfolio Challenge,” which ended May 25, as players put their stock-picking skills to the test. But now New Jersey-based CNBC says it is investigating “unusual trading in violation of contest rules among some of the 20 finalists” after contestants alerted the network that some of their rivals may have been gaming the trading system.
Bypassing Security Controls ?
Business Logic Flaws Categorized in 2 Groups

Group 1: Exploit flaws in enforcement of business logic for transactions and the data, examples include:
Weak enforcement of business logic such as workflows and steps required by transactions (e.g. A, B, C not A to C directly)
Insufficient parameters validation (e.g. priceID, roleIDs, userID)
Committing to transactions without doing all checks required first

Group 2: Exploit weaknesses in security controls whose function is protect the business rules, examples include:
Mis-configurations of Role Base Access Controls (RBAC) policy rules,
Password reset flaws, username recovery flaws,
Security controls failing insecurely,
Insufficient anti-automation attacks defenses
Vulnerability Analysis of Business Logic Flaws
BLA From Risk Analysis Perspective

- Threats
  - Some threat agent (human, non-human) and/or adverse condition that target the application logic to cause a negative impact to the business and the customers

- Attacks
  - Realize the threat to business logic cause a negative impact, includes different ways for an attacker to conduct business logic attacks by exploiting one or more vulnerabilities and logic flaws

- Application vulnerabilities
  - Are weaknesses in the application that can be exploited by a threat and cause a negative impact to the application.
Root Causes Of Business Logic Flaws

- **Security Design Flaws**
  - Caused by lack of security requirements, poor knowledge, lack of security architecture design reviews
  - Cannot be identified by security tools alone since are logical vulnerabilities and require manual threat analysis/threat modeling

- **Security Coding Errors**
  - Coding bugs that result in vulnerabilities
  - Can be identified with source code analysis tools and manual code reviews

- **Security Mis-configurations**
  - Mis-configuration for application security policies and business rules
  - Can be identified through change control processes

Business Logic Attack, Mostly Exploit Security Flaws in Design and Security Mis-Configurations
Vulnerabilities Potentially Exploited By BLAs


**Summary**

<table>
<thead>
<tr>
<th>Weakness Prevalence</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Security bypass</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Remediation Cost</th>
<th>Ease of Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low to Medium</td>
<td>Moderate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack Frequency</th>
<th>Attacker Awareness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Often</td>
<td>High</td>
</tr>
</tbody>
</table>

**Discussion**

Suppose you're hosting a house party for a few close friends and their guests. You invite everyone into your living room, but while you're catching up with one of your friends, one of the guests raids your fridge, peeks into your medicine cabinet, and ponders what you've hidden in the nightstand next to your bed. Software faces similar authorization problems that could lead to more dire consequences. If you don't ensure that your software's users are only doing what they're allowed to, then attackers will try to exploit your improper authorization and exercise unauthorized functionality that you only intended for restricted users.
1: BLAs Exploiting Authorization Flaws

- **BUSINESS LOGIC ATTACKS:**
  - Attackers access web resources not restricted by role, simply changes the workflow/URL to a privileged page using forceful browsing
  - Attackers change the parameters of a business transaction such as the price of goods purchased to be charged a cheaper price

- **FLAWS:**
  - INSUFFICIENT AUTHORIZATION (WASC-02),
  - FAILURE TO RESTRICT URL ACCESS (OWASP A7),
  - IMPROPER ACCESS CONTROL (SANS-CWE-285)

- **ROOT CAUSES:**
  - Lack of granular enforcement of authorization rules through policy such as Role Base Access Controls (RBAC)
  - Business rules enforced using client side parameters instead of server side logic
2: BLAs Exploiting Authentication Flaws

**BUSINESS LOGIC ATTACKS:**
- Attackers guess questions in challenge/question authentication (e.g. account creation, change password, recover password, this includes KBA, Knowledge Based Authentication)
- **Attackers replay the session** such as a valid sessionID to logon in the application after previous logout

**FLAWS:**
- INSUFFICIENT AUTHENTICATION (WASC-01),
- BROKEN AUTHENTICATION AND SESSION MANAGEMENT (OWASP A3),
- MISSING AUTHENTICATION FOR CRITICAL FUNCTION (CWE 306)

**ROOT CAUSES:**
- Design flaws for password reset transactions
- Easily guessable questions
- **Session management issues** such as lack of single logout across applications-tiers
3: BLAs Exploiting Mis-Configurations

**BUSINESS LOGIC ATTACKS:**

- Attackers exploit mis-configuration of access control policy to exploit fail open-insecure conditions, unauthorized access to resources, bypass of authentication and RBAC, information disclosure through errors

- Attackers bypass detection since transaction and security events are not logged so the attack cannot be audited/investigated

**FLAWS:**

- SERVER MISCONFIGURATION (WASC-14),
- SECURITY MISCONFIGURATION (OWASP A6)

**ROOT CAUSES:**

- Configuration management changes not tested for enforcement of roles and permissions
- Logging does not cover validations x transaction x user
4: BLAs Exploiting Insufficient Anti-Automation

**BUSINESS LOGIC ATTACKS:**

- **Automatic injection of web pages** (e.g. forms/Frames) in application workflows to collect PII (e.g. Zeus Trojans) to commit fraud
- **Automated validation of credit card data** through the application exploiting error and exception handling flaws
- **Spam of account registrations** to flood back-office processes
- **Denial of services to customers by locking accounts by failing logins** through automation locking and by flooding of call center for unlocking requests

**FLAWS:**

- **INSUFFICIENT ANTI-AUTOMATION (WASC-21)**

**ROOT CAUSES:**

- Lack of detective control for automation (e.g. CAPTCHA, automated intrusion detection) to protect transactions
5: BLAs Exploiting Insufficient Process Controls

**BUSINESS LOGIC ATTACKS:**
- Fraudster bypasses validations checks for performing transactions such as by altering the flow (e.g. shipping for goods not being purchased)
- Attacker learns to exploit business logic from client code (e.g. Web 2.0 applications)
- Attacker learns how to game the system from the way the system responds to input data

**VULNERABILITY**
- INSUFFICIENT PROCESS CONTROLS (UNCLASSIFIED)

**ROOT CAUSES:**
- Insufficient enforcement of transaction validations performed at different stages of the business transaction before committing to it
- Lack of data validations at different tiers of the application architecture (e.g. application and messaging)
- Lack of Out of Band (OOB) validations and call backs
- Business logic exposed to clients
Identification and Mitigation of Business Logic Flaws
Business Logic In Web Application Architectures

**Presentation Tier**
Represents the top most level of the application. The purpose of this tier is to translate commands from the user interface into data for processing to other tiers and present back the processed data.

**Logic Tier**
This layer processes commands and makes decisions based upon the application business logic. It also moves and processes data between the presentation and the data tier.

**Data Tier**
Is the layer responsible for data storage and retrieval from a database or file system. Query commands or messages are processed by the DB server, retrieved from the datasource and passed back to the logical tier for processing before being presented to the user.

Not All Business Logic Resides on the Application Server!

Beware of Web 2.0 Apps that include business logic client side (e.g. AJAX, Widgets, Mashups)

Beware of Flaws in Integration of Business Logic with Server Components
Security Process for Preventing Business Logic Flaws in Applications

1. **Require every application to document business logic** with data flows for transactions and the access control matrix used.

2. **Design the application as business logic abuse resistant**, including process validations and controls assuming that the application business logic can be abused.

3. **Use application threat modeling** to identify design flaws in business logic. Analyze abuse of business logic and security controls with use misuse cases and transaction analysis.

4. **Security test (manually) for business logic flaws/vulnerabilities such as** OWASP 3, 4, 8, WASC 1, 2, 14, 21, and SANS-25-CWE 285, 306.

5. **Create specific security tests for abuse of business logic** by deriving them from the use-abuse cases and transaction/data flow analysis performed during threat modeling.

6. **Analyze risks and apply countermeasures** to mitigate likelihood and impact of business logic attacks.
Identification Of Security Flaws: Application Threat Modeling Process

1. Define Objectives
   - Identify Business Objectives
   - Identify Security Objectives
   - Business Impact Analysis

2. Scope Definition
   - Understanding application boundaries
   - Identify APIs & communication protocols
   - Data Flow Diagramming Exercises or Software Modeling
   - Identify process boundaries that affect application environment

3. Application Decomposition
   - Use Cases
   - Actors / Assets
   - Data Sources / APIs

4. Threat Analysis
   - Probabilistic attack scenarios
   - Qualify attack scenarios
   - Define countermeasures where appropriate

5. Vulnerability Mapping
   - Technology vulnerabilities mapped to application components
   - Use case to misuse case mapping
   - Vulnerabilities with technical requirements
   - Business logic vulnerabilities

6. Attack Tree
   - Use to Mitigate Case Mapping
   - Asset to vulnerability
   - Vulnerability / Exploit Mapping

7. Risk & Impact Analysis
   - ID risk mitigation strategies
   - Integrate metrics for financial impact
   - Qualify / quantify business impact

OWASP
Data Flow Diagramming

- User's Browser
  - HTTPS/Request
  - Responses
- Web Servers
  - HTTPS Requests To Application Framework
  - Responses
- Application Server
  - Web-service Calls SOAP/XML
  - Web Service SOAP Response
  - JDBC Call
  - JDBC Responses
  - Elevation Of Privileges/RBAC Misconfigurations
  - Spoofing And Tampering SQL Queries
- Database Server
  - Queried SQL Query Call Data
  - Data
- Enterprise Service Bus
  - Message Queue Call
  - Message Call Response
- Legacy Mainframe
  - Message Calls
- DMZ-Boundary
- Spoofing And Tampering Web Service Calls
  - Web-service Calls SOAP/XML
- Spoofing And Tampering Message Calls
  - Message Queue Call
- Threats to Application Business Logic
Security Flaws Identification With Use And Abuse Cases: Password Reset

User

- Register Challenge Questions

- Select Password Reset

- Enter User ID And Submit

- Receive Temporary Password in Email

- Enter User ID And One Time Password

- Answer Any of 2 Secret Questions

- Create New Password

- Lockout After N Attempts to Answer

- Use Shared Secrets, High Entropy Questions

- Enumerate Valid Answers

Compromised User’s Email

- Mitigates

User Email Verified

- Threatens

Random OTP, 6 char & numbers, last 15 minute

Guess OTP

- Mitigates

Guess Answers

- Mitigates

Hacker/Malicious User

- Threatens

Receive Temporary Password in Email

- Includes

Receive Temporary Password in Email

- Includes

Receive Temporary Password in Email

- Includes
Identification of Business Logic Flaws Using Transaction Analysis: Shopping Cart

1. Shopping Start
2. Shopping/Browse For Items in the Catalogue
3. Add Item to Shopping Cart
4. Add Check Out Item
5. Login or Sign up
6. Continue Shopping
7. Is user logged in?
   - YES: Proceed to Check out
   - NO: Business Logic Attack
8. Select Items
9. Calculate Total
10. Select Type Of Shipping
11. Enter Shipping Address
12. Checkout Credit Card
13. Charge Passed
   - YES: Order Thank You
   - NO: Business Logic Attack
14. Items in the Shopping Cart?
15. Checkout View Cart
16. Charge Passed
17. Order Thank You
Shopping Cart BLA Example

Catalogue Price: $27.99

Charged Price: $.99
Testing For Occurrence of Business Logic Flaws

- Main objective is to test that the application business rules cannot be altered by business logic attacks
- Require testers to write NEGATIVE test cases and scripts to identify potential exploits of business logic flaws during Q/A test validation cycles. Examples include:
  1. Trying to bypass of user validations and prerequisite checks for a transaction,
  2. Trying to bypass multi factor authentication in a transaction,
  3. Trying to force a transaction and access high privileged resources logging as low privilege user,
  4. Tampering with business logic parameters during a request to try to access resources,
  5. Replaying session tokens after logouts to try to log back to the application,
  6. Trying to force the application to fail in unsecure conditions such as fail open or as un-handled exceptions
  7. Trying to alter price of items and validate if they can be added,
  8. Trying to abuse registration, account openings/applications with automation scripts
# Checking Configuration of Security And Business Rules

<table>
<thead>
<tr>
<th>Role1</th>
<th>Role2</th>
<th>Role3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource1</td>
<td>Deny</td>
<td>Grant</td>
</tr>
<tr>
<td>Resource2</td>
<td>Grant</td>
<td>Deny</td>
</tr>
</tbody>
</table>

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AccessControlPolicy>
  <AccessControlRules>
    <AccessControlRule
      name="Grant"
      description="Access is always granted"
      class="org.owasp.esapi.accesscontrol.AlwaysTrueACR">
    </AccessControlRule>
    <AccessControlRule
      name="Deny"
      description="Access is always denied"
      class="org.owasp.esapi.accesscontrol.AlwaysFalseACR">
    </AccessControlRule>
    <AccessControlRule
      name="SpecialCaseAccessControlRule"
      description="Access depends on the value of the policy parameter: isTrue"
      class="org.owasp.esapi.accesscontrol.policy.EchoDynaBeanPolicyParameterACR">
      <Parameters>
        <Parameter name="isTrue" type="Boolean" value="true"/>
      </Parameters>
    </AccessControlRule>
  </AccessControlRules>
</AccessControlPolicy>
```
Possible Countermeasures Against BLAs

- **Deterrent controls**
  - Anti-automation (e.g. CAPTCHA, logic puzzles)

- **Preventive controls**
  - Strong authentication and authorization of transactions (e.g. ESAPI)
  - Secure password reset and userID reminder processes
  - Strong business process validation/checks for transactions (e.g. use Out Of Band)
  - Data validation/filtering of transaction parameters (e.g. ESAPI)
  - Secure session management such as the SessionIDs used in business transactions

- **Detective controls**
  - Application layer detection rules for BLA patterns (e.g. ESAPI IDS)
  - Web Application Firewall (WAF) rules (e.g. ESAPI WAF)
  - Fraud monitoring and detection rules (e.g. Fraud Detection)
  - Logging and alerts of business transaction events as well as related security events
QUESTIONS & ANSWERS
Thanks for listening, further references

- Designing a Framework Method for Secure Business Application Logic Integrity in e-Commerce Systems
- Seven Business Logic Flaws That Put Your Website At Risk
- Testing for business logic (OWASP-BL-001)
- Get rich or die trying, “Making money on the web, the black hat way”
Further references con’t

- OWASP Top Ten Project

- The WASC Threat Classification v2.0
  - [http://projects.webappsec.org/Threat-Classification](http://projects.webappsec.org/Threat-Classification)

- CWE/SANS TOP 25 Most Dangerous Coding Errors

- OWASP Application Threat Modeling

- OWASP EASPI

- OWASP Testing Project