Top Ten - 2010 rc1

The Ten Most Critical Web Application Security Risks
Risks to your business processes & info systems
OWASP Top 10 Risk Rating Methodology

Exploitability | Prevalence | Detectability | Impact
---|---|---|---
DIFFICULT | UNCOMMON | DIFFICULT | 
AVERAGE | COMMON | AVERAGE | MODERATE
EASY | WIDESPREAD | EASY | SEVERE
VERY WIDESPREAD |
Example for Cross Site Scripting (XSS)

Score = Weighted risk rating
= Average of Exploitability, Prevalence and Detectability multiplied by Impact
= \((2 + 1 + 1)/3 \times 2\)
= 2.6
Evaluate your own business risks

Use OWASP’s Risk Rating Methodology
http://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

- Step 1: Identifying a risk
- Step 2: Factors for estimating likelihood
- Step 3: Factors for estimating impact
- Step 4: Determining severity of the risk
- Step 5: Deciding what to fix
- Step 6: Customizing your risk rating model
A10 - Insufficient Transport Layer Protection

**Transmitting sensitive data insecurely**
- Failure to identify all sensitive data
- Failure to identify all the places that this sensitive data is sent
  - On the web, to backend databases, to business partners, internal communications
- Failure to properly protect this data in every location

**Typical Impact**
- Attackers access or modify confidential or private information
  - e.g., credit cards, health care records, financial data (yours or your customers)
- Attackers extract secrets to use in additional attacks
- Company embarrassment, customer dissatisfaction, and loss of trust
- Expense of cleaning up the incident
- Business gets sued and/or fined
A9 - Insecure Cryptographic Storage

Storing sensitive data insecurely

- Failure to identify all sensitive data
- Failure to identify all the places that this sensitive data gets stored
  - Databases, files, directories, log files, backups, etc.
- Failure to properly protect this data in every location

Typical Impact

- Attackers access or modify confidential or private information
  - e.g., credit cards, health care records, financial data (yours or your customers)
- Attackers extract secrets to use in additional attacks
- Company embarrassment, customer dissatisfaction, and loss of trust
- Expense of cleaning up the incident, such as forensics, sending apology letters, reissuing thousands of credit cards, providing identity theft insurance
- Business gets sued and/or fined
A8 - Unvalidated Redirects and Forwards

Web application redirects are very common

- And frequently include user supplied parameters in the destination URL
- If they aren’t validated, attacker can send victim to a site of their choice

Forwards (aka Transfer in .NET) are common too

- They internally send the request to a new page in the same application
- Sometimes parameters define the target page
- If not validated, attacker may be able to use unvalidated forward to bypass authentication or authorization checks

Typical Impact

- Redirect victim to phishing or malware site
- Attacker’s request is forwarded past security checks, allowing unauthorized function or data access
A7 - Failure to Restrict URL Access

How do you protect access to URLs (pages)?
- This is part of enforcing proper “authorization”, along with A4 – Insecure Direct Object References

A common mistake ...
- Displaying only authorized links and menu choices
- This is called presentation layer access control, and doesn’t work
- Attacker simply forges direct access to ‘unauthorized’ pages

Typical Impact
- Attackers invoke functions and services they’re not authorized for
- Access other user’s accounts and data
- Perform privileged actions
A6 - Security Misconfiguration

Web applications rely on a secure foundation
- All through the network and platform
- Don’t forget the development environment

Is your source code a secret?
- Think of all the places your source code goes
- Security should not require secret source code

CM must extend to all parts of the application
- All credentials should change in production

Typical Impact
- Install backdoor through missing network or server patch
- XSS flaw exploits due to missing application framework patches
- Unauthorized access to default accounts, application functionality or data, or unused but accessible functionality due to poor server configuration
A5 - Cross-Site Request Forgery (CSRF)

Cross Site Request Forgery

- An attack where the victim’s browser is tricked into issuing a command to a vulnerable web application
- Vulnerability is caused by browsers automatically including user authentication data (session ID, IP address, Windows domain credentials, ...) with each request

Imagine...

- What if a hacker could steer your mouse and get you to click on links in your online banking application?
- What could they make you do?

Typical Impact

- Initiate transactions (transfer funds, logout user, close account)
- Access sensitive data
- Change account details
A4 - Insecure Direct Object References

How do you protect access to your data?

- This is part of enforcing proper “Authorization”, along with A7 – Failure to Restrict URL Access

A common mistake ...

- Only listing the ‘authorized’ objects for the current user, or
- Hiding the object references in hidden fields
- ... and then not enforcing these restrictions on the server side
- This is called presentation layer access control, and doesn’t work
- Attacker simply tampers with parameter value

Typical Impact

- Users are able to access unauthorized files or data
A3 - Broken Authentication & Session Management

HTTP is a “stateless” protocol

- Means credentials have to go with every request
- Should use SSL for everything requiring authentication

Session management flaws

- SESSION ID used to track state since HTTP doesn’t
  - and it is just as good as credentials to an attacker
  - SESSION ID is typically exposed on the network, in browser, in logs, ...

Beware the side-doors

- Change my password, remember my password, forgot my password, secret question, logout, email address, etc...

Typical Impact

- User accounts compromised or user sessions hijacked
A2 - Cross-Site Scripting (XSS)

**Occurs any time...**
- Raw data from attacker is sent to an innocent user’s browser

**Raw data...**
- Stored in database
- Reflected from web input (form field, hidden field, URL, etc...)
- Sent directly into rich JavaScript client

**Virtually every web application has this problem**
- Try this in your browser – javascript:alert(document.cookie)

**Typical Impact**
- Steal user’s session, steal sensitive data, rewrite web page, redirect user to phishing or malware site
- Most Severe: Install XSS proxy which allows attacker to observe and direct all user’s behavior on vulnerable site and force user to other sites
A1 - Injection

Injection means...
- Tricking an application into including unintended commands in the data sent to an interpreter

Interpreters...
- Take strings and interpret them as commands
- SQL, OS Shell, LDAP, XPath, Hibernate, etc...

SQL injection is still quite common
- Many applications still susceptible (really don’t know why)
- Even though it’s usually very simple to avoid

Typical Impact
- Usually severe. Entire database can usually be read or modified
- May also allow full database schema, or account access, or even OS level access
Risk calculation
How to detect if you are vulnerable
Examples attacks
How to prevent it
References
Additional advice

- What’s next for developers
- What’s next for verifiers
- ?
- Notes about risk
## Summary of changes 2007 to 2010 rc1

<table>
<thead>
<tr>
<th>OWASP Top 10 – 2007 (Previous)</th>
<th>OWASP Top 10 – 2010 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2 – Injection Flaws</td>
<td>A1 – Injection</td>
</tr>
<tr>
<td>A1 – Cross Site Scripting (XSS)</td>
<td>A2 – Cross Site Scripting (XSS)</td>
</tr>
<tr>
<td>A7 – Broken Authentication and Session Management</td>
<td>A3 – Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A4 – Insecure Direct Object Reference</td>
<td>A4 – Insecure Direct Object References</td>
</tr>
<tr>
<td>A5 – Cross Site Request Forgery (CSRF)</td>
<td>A5 – Cross Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>&lt;was T10 2004 A10 – Insecure Configuration Management&gt;</td>
<td>A6 – Security Misconfiguration (NEW)</td>
</tr>
<tr>
<td>A10 – Failure to Restrict URL Access</td>
<td>A7 – Failure to Restrict URL Access</td>
</tr>
<tr>
<td>&lt;not in T10 2007&gt;</td>
<td>A8 – Unvalidated Redirects and Forwards (NEW)</td>
</tr>
<tr>
<td>A8 – Insecure Cryptographic Storage</td>
<td>A9 – Insecure Cryptographic Storage</td>
</tr>
<tr>
<td>A9 – Insecure Communications</td>
<td>A10 - Insufficient Transport Layer Protection</td>
</tr>
<tr>
<td>A3 – Malicious File Execution</td>
<td>&lt;dropped from T10 2010&gt;</td>
</tr>
<tr>
<td>A6 – Information Leakage and Improper Error Handling</td>
<td>&lt;dropped from T10 2010&gt;</td>
</tr>
</tbody>
</table>
The End