Defending your Web Applications from Attack:
Current Web-Based Threats, Resources & Tools

Presenter: Damira Pon, UAlbany
Who am I?

Introduction
Damira Pon is a faculty in the Digital Forensics Program and a part of the School of Business and School of Criminal Justice at the University at Albany, State University of New York. She is currently an ISACA Hudson Valley Chapter Board Member & Academic Advocate; she also acts as one of the faculty co-advisors for the UAlbany ISACA student group and Digital Forensics Association. Prior to becoming a faculty member in 2015, she was a Senior Research Analyst at the NYS Center for Information Forensics and Assurance, where she worked for more than 10 years. She has participated in several NYS Cyber Security Conference plenary sessions and workshops which included live hacking demonstrations and workshops on incident analysis and risk assessment. She has wide range of experience in online and blended education in information security and forensics, information security policy and compliance (business continuity / disaster recovery, information classification, risk assessment), creation of standard operating procedures for law enforcement in digital forensics, crime mapping related to Internet Crimes against Children (ICAC), data analysis (including cyber incident and log analysis), and penetration testing for IT security audits.
5 Effective Habits of Highly Effective Application Security Leaders

1. Align Security as a Business Enabler Throughout the Organization
2. Engage Developers with Security
3. Correct Tools are in Place and Used Correctly
4. Part of Boardroom Discussions
5. Make Concerted Effort to Exchange Information between Teams
What will we be talking about today?

Agenda

- Web Security Threat Environment
- Web Hacking Clues
- Best Practices
- Resources & Tools
- Conclusion
- Questions
Why should we care about web security?

Web Security Threat Environment
Information Security Classification

CIA3+

1. Confidentiality
   • Violation: Loss of PPI (Personal Private Information) or PPSI (Personal Private Sensitive Information)
   • e.g., data breaches

2. Integrity
   • Violation: Destruction or Modification of Code
   • e.g., code injection

3. Availability
   • Violation: Prevention of access to resources
   • e.g., DDoS or DoS (Distributed / Denial-of-Service)
4. Authentication
   • Violation: Access without correct credentials
     e.g., brute-force password attacks to gain access

5. Authorization
   • Violation: Access to unauthorized resources
     e.g., privilege escalation

6. Non-Repudiation
   • Violation: Origin of activities cannot be verified
     e.g., MiTM (Man-in-the-Middle) / session hijacking
| A1 | Injection | Injection flaws, such as SQL, OS, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker’s hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data. |
| A2 | Broken Authentication and Session Management | Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, session tokens, or exploit other implementation flaws to assume other users’ identities. |
| A3 | Cross-Site Scripting (XSS) | XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation or escaping, XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites. |
| A4 | Insecure Direct Object References | A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data. |
| A5 | Security Misconfiguration | Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. All these settings should be defined, implemented, and maintained as many are not shipped with secure defaults. This includes keeping all software up to date. |
| A6 | Sensitive Data Exposure | Many web applications do not properly protect sensitive data, such as credit cards, tax ids, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct identity theft, credit card fraud, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser. |
| A7 | Missing Function Level Access Control | Generally web applications verify function level access rules before making that functionality visible in the UI. However, applications need to perform same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to force requests in order to access unauthorized functionality. |
| A8 | Cross-Site Request Forgery (CSRF) | A CSRF attack forces a logged-on victim’s browser to send a forged HTTP request, including the victim’s session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim’s browser to generate requests the vulnerable application thinks are legitimate requests from the victim. |
| A9 | Using Components with Known Vulnerabilities | Vulnerable components, such as libraries, frameworks, and other software modules almost always run with full privilege. So, if exploited, they can cause serious data loss or server takeover. Applications using these vulnerable components may undermine their defenses and enable a range of possible attacks and impacts. |
| A10 | Unvalidated Redirects and Forwards | Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages. |
2015 Web Application Attack Report (WAAR)
Imperva Annual Report on Web-Based Attacks

Key

• SQLi (SQL Injection Attack)
• XSS (Cross Site-Scripting Attack)
• RCE (Remote Command Execution Attack)
• DT (Directory Traversal Attack)
• HTTP (Malicious HTTP Requests)
• RFI (Remote File Inclusion Attack)
• Spam (Comment Spam Attack)
• FU (File Upload Attack)
2015 Web Application Attack Report (WAAR)

Key Findings #1: Threat Growth

Threats are Increasing

- SQLi (up 200%)
- XSS (up 150%)
- Relation to WordPress

Source: Imperva
2015 Web Application Attack Report (WAAR)
Key Findings #1: Threat Growth, Cont’d.

Attacks More Widespread
• 75% of applications exposed to attacks of each type
• Shellshock RCE had attempts on 100% of applications
  ▪ Made public Sept. 2014
  ▪ Similar to Heartbleed (quick turnaround, automation, large)

Source: Imperva
Mitigation Classes

- Detect-by-reputation effective
  - 78% of alerts
  - 100,000 malicious requests blocked for an application / mo.
- Blocking based on IP addresses
- Automation of attacks from organized crime syndicates

Source: Imperva
2015 Web Application Attack Report (WAAR)

Key Findings #3: Attack Targets

Attack Targets

• CMS attacked 3x more
  ▪ WordPress attacked 3.5x more

• WordPress targeted 7x more for Spam / RFI

• XSS 10x more popular for health-related applications

Source: Imperva

Source: WordPress
How would I know I am being attacked?

Web Hacking

Clues
Web Hacking Clues
Web Analytics for Attack Detection

• Analytics useful for identifying hacking
  ▪ IP geolocation (increases in uncommon cities)
  ▪ Pages / Visit (closer to 0 than avg.)
  ▪ Time on Site (closer to 0 than avg.)
  ▪ % of New Visitors (closer to 100% than avg.)
  ▪ Bounce Rate (closer to 100% than avg.)
Web Hacking Clues

Using Alerts with Web Analytics

• Set Alerts (e.g., Google Intelligence)
  • Can send email alerts (as-it-happens)
  • Check for suspicious activity & common spam terms
• Be wary of analytics getting “tainted”
  ▪ add filters for hostname in custom filter with filter pattern: domainname\.com
Web Hacking Clues

Web Logs

• Ensure collection & storage of logs
  ▪ GET with successful HTTP status code 200
  ▪ Suspicious user agents
  ▪ Multiple unsuccessful logins from same IP
  ▪ IP testing out multiple usernames within short amount of time
  ▪ Privileged access and activities
Honeypots can:
- Attract attackers
- Examine threat actors, tools & techniques

Specialized honeypots for web:
- Kippo (SSH honeypot for logging brute force attacks)
- Glastoph (web application honeypot good for local file inclusion & SQL injection)
- Dionaea (malware collection honeyot which emulates vulnerabilities)
- Thug (client-side honeypot emulates web browser & interacts with malicious websites)
What should you do?

Best Practices
Best Practices
Server Configuration Files

- Know your configuration files
  - Apache (.htaccess)
  - Nginx (nginx.conf)
  - Microsoft IIS (web.config)

- Check site using (Website Details):
  https://sitecheck.sucuri.net

- Limit information viewed
  - Directory browsing
  - Image hotlinking
  - Sensitive files & execution of php
Best Practices

Server Settings & Applications

• Avoid default settings (especially for CMS)
• Host one site per “container”
• Do not use default usernames & passwords
• Select extensions / plugins carefully
  ▪ Dates (last updated, age, # of installs)
  ▪ Download from legitimate service
  ▪ Don’t use pirated “free” versions
• Ensure system, applications & plugins are up-to-date & only necessary software is installed
Best Practices

SSL

• Use https
  • Helps prevent MiTM attacks & eavesdropping
    ▪ Can result in changing form action
    ▪ Can inject javascript for keystroke logging of input
  • Encrypts communications from web server & browser
  • Especially important for forms with personal or sensitive information
Best Practices

File Permissions & Access Privileges

• Permissions
  ▪ Read (4)
  ▪ Write (2)
  ▪ Execute (1)

• Users
  ▪ User (owner)
  ▪ Group
  ▪ Other (public)

• Be careful – don’t just use 666 or 777 permissions!
• Look at admin / log files / directory structure
• Only give access to those who need it (least privileged)
• Review logs regularly
Best Practices

Passwords

• Use complex & long passwords
  ▪ Include special characters, upper/lowercase alphabet & numbers and at least 12 characters
  ▪ User feedback for password strength

• Avoid password reuse
  ▪ Encourage use of password managers

• Reset after compromise

• Prevent brute-force attacks
  ▪ Captchas
  ▪ Scripts & alerts based on unsuccessful logins
Backup site securely
- Dangers due to unpatched & earlier versions
- Different server location than website (hardware & location)
- Automation
- Redundant (mirrored & in multiple location)
- Tested regularly
Cookies

- Cookies used to maintain sessions
  - Can be targeted with XSS and eavesdropping
  - Secure flag (Prevents sending over HTTP)
  - HTTPOnly flag (only sent in HTTP headers)
  - Session or with expiration (avoid long expiration dates)
  - DO NOT store sensitive information (store on server & store session identifier on cookie) - Check if your framework does not do this by default
Others framing your site – prevent clickjacking
- X-Frame-Options
- Parameters: deny, sameorigin, allow-from

iframes sandbox (HTML5)
- Allows for unique origin, blocking of form submission, script execution, APIs, links from targeting browser contexts, content from plugins, content from navigating top-level browsing content
- Fully supported in IE10, Chrome 4.0 Firefox 7.0 Safari 5.0 and Opera 15.0
- `<iframe sandbox="value">`
- Parameters: allow-forms, allow-pointer-lock, allow-popups, allow-same-origin, allow-scripts, allow-top-navigation
Best Practices

Open Redirects, User Feedback, & Mixed Content

• Avoid URL open redirects
  ▪ Restrict &url to whitelist of predefined websites & add signature parameter to match URL

• Change user authentication feedback
  ▪ Include in comments something like: bad password, username
  ▪ Instead of same HTTP status error, give a response page or change what occurs

• Prevent mixed content by using https
  ▪ Check for http:// in HTML, sourced javascript / css, requests made by objects or create relative links to protocol
Best Practices
Secure Coding – Forms

- Forms
  - Perform input validation specifically check for <, >, ‘, “, & and translate to HTML e.g., &lt; &quot; &amp; See - htmlspecialchar() or framework
  - Use POST method when including connections to database in combination with “https://” action
  - GET puts input in URL
  - Use hidden form field / URL parameter to include token with request (tied to user cookie) – prevents CSRF
Best Practices

Secure Coding – Content Security Policy

- **Content-Security-Policy header**
  - Allows you to specify what is allowed
  - By default inline scripts / styles, and eval() disabled
  - UserCSP (firefox add-on) or NWebsec

```html
<content-Security-Policy enabled="true">
  <default-src self="true" />
  <script-src unsafeInline="true" unsafeEval="true" self="true">
    <add source="https://www.google.com" />
    <add source="https://www.google-analytics.com" />
    <add source="https://cdnjs.cloudflare.com" />
  </script-src>
  <style-src unsafeInline="true" self="true">
    <add source="https://cdnjs.cloudflare.com" />
  </style-src>
  <img-src self="true">
    <add source="https://az594751.vo.msecnd.net"/>
    <add source="https://www.google.com"/>
    <add source="https://www.google-analytics.com"/>
  </img-src>
  <font-src>
    <add source="https://cdnjs.cloudflare.com"/>
  </font-src>
  <object-src none="false"/>
  <media-src none="false"/>
  <frame-src none="false"/>
  <connect-src none="false"/>
  <frame-ancestors none="false"/>
  <report-uri enableBuiltInHandler="true"/>
</content-Security-Policy>
```

Best Practices
Stay Informed & Be Aware

• New vulnerabilities, threats & controls
  ▪ Software & firmware updates
• News
  ▪ Twitter
  ▪ Vulnerability Databases
  ▪ Application Websites
• Vulnerability Scanning
• Penetration testing & red team exercises
• Awareness training
CSC 1: Inventory of Authorized and Unauthorized Devices
CSC 2: Inventory of Authorized and Unauthorized Software
CSC 3: Secure Configurations for Hardware and Software on Mobile Devices, Laptops, Workstations, and Servers
CSC 4: Continuous Vulnerability Assessment and Remediation
CSC 5: Controlled Use of Administrative Privileges
CSC 6: Maintenance, Monitoring, and Analysis of Audit Logs
CSC 7: Email and Web Browser Protections
CSC 8: Malware Defenses
CSC 9: Limitation and Control of Network Ports, Protocols, and Services
CSC 10: Data Recovery Capability

http://www.cisecurity.org/critical-controls/
Resources & Tools
CIS Critical Security Controls 11-20

CSC 11: Secure Configurations for Network Devices such as Firewalls, Routers, and Switches
CSC 12: Boundary Defense
CSC 13: Data Protection
CSC 14: Controlled Access Based on the Need to Know
CSC 15: Wireless Access Control
CSC 16: Account Monitoring and Control
CSC 17: Security Skills Assessment and Appropriate Training to Fill Gaps
CSC 18: Application Software Security
CSC 19: Incident Response and Management
CSC 20: Penetration Tests and Red Team Exercises

http://www.cisecurity.org/critical-controls//
What’s useful?

Resources & Tools
Resources & Tools
Vulnerability Scanning & Testing

• General
  ▪ Nessus, OpenVAS, QualysGuard, Acunetix, Retina, nmap / zenmap
  ▪ [http://sectools.org/tag/vuln-scanners/](http://sectools.org/tag/vuln-scanners/)

• Web Vulnerability Scanners
  ▪ BurpSuite, Nikto, w3af, Paros proxy, WebScarab, sqlmap, skipfish, Acunetix WVS, AppScan, Netsparker, HPWebInspect, Wikto, Samurai Web Testing Framework, Firebug
  ▪ [http://sectools.org/tag/web-scanners/](http://sectools.org/tag/web-scanners/)
Resources & Tools
Vulnerability Scanning & Testing

• System Specific
  ▪ Drupal Security Scan
    https://hackertarget.com/drupal-security-scan/
  ▪ WPScan Wordpress Plugin & Vulnerability Database
    https://wpvulndb.com/
  ▪ Joomla Security Scan
    https://hackertarget.com/joomla-security-scan/
Resources & Tools

• Top 10 Lists
• Cheat Sheets
• Guides
  ▪ Developers Guide
  ▪ Code Review Guide
  ▪ Testing Guide
• ZAP Proxy
• Web Hacking Incident Database

https://www.owasp.org
## Resources & Tools
### Top 10 Scanning

<table>
<thead>
<tr>
<th>WEB APPLICATION RISK</th>
<th>SECURITY UTILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1: Injection</td>
<td>SQL Inject Me and Zed Attack Proxy (ZAP)</td>
</tr>
<tr>
<td>A2: Broken Authentication and Session Management</td>
<td>ZAP</td>
</tr>
<tr>
<td>A3: Cross-Site Scripting (XSS)</td>
<td>ZAP</td>
</tr>
<tr>
<td>A4: Insecure Direct Object References</td>
<td>HTTP Directory Traversal Scanner, Burp Suite and ZAP</td>
</tr>
<tr>
<td>A5: Security Misconfiguration</td>
<td>OpenVAS and WATOBO</td>
</tr>
<tr>
<td>A6: Sensitive Data Exposure</td>
<td>Qualys SSL Server Test</td>
</tr>
<tr>
<td>A7: Missing Function Level Access Control</td>
<td>OpenVAS</td>
</tr>
<tr>
<td>A8: Cross-Site Request Forgery (CSRF)</td>
<td>Tamper Data (Samurai WTF), WebScarab or ZAP</td>
</tr>
<tr>
<td>A9: Using Components with Known Vulnerabilities</td>
<td>OpenVAS</td>
</tr>
<tr>
<td>A10: Unvalidated Redirects and Forwards</td>
<td>ZAP</td>
</tr>
</tbody>
</table>

Resources & Tools
AmanHardikar.com Mindmaps

- Web Application Testing
- Code Review
- Useful Browser Plugins
- Infrastructure Testing
- Hacking Practice Lab

http://amanhardikar.com/mindmaps.html
Conclusion

- Web & Security need to work together
- Attacks are happening all the time
- Environment constantly changing – new vulnerabilities, threats, systems & controls
- Web applications must be protected
- Tools and resources are available – there is a lot to know in this area!
What would you like to ask?

Questions & Thank You!

Also reach me offline by email: dpon@albany.edu or Google Voice: (518) 227-0088
References

Additional Sources Used

- https://www.checkmarx.com/2015/01/26/application-security-leaders/
- http://www.seerinteractive.com/blog/detect-hacking-attempts-with-google-analytics/
- https://blog.kissmetrics.com/protect-analytics-from-hacking/
- https://www.branded3.com/blog/how-to-use-google-alerts-to-find-out-if-your-site-gets-hacked/
- https://blog.sucuri.net/2015/06/10-tips-to-improve-your-website-security.html
- https://blog.sucuri.net/2015/04/how-to-create-a-website-backup-strategy.html
- https://air.mozilla.org/web-security-training/