Server Security

Todd King

Mark Rose

Sean Hardman

Potential Issues*

- Insufficient compartmentalization of different web applications - in particular, different process running as the same user.
- Insufficient sanitizing of inputs (XSS, SQL injection, etc.)
- Excessive privileges for server processes (for example, running as root).
- Extraneous open ports (for example, database server or administrative interfaces)
- Out of date third party libraries in web application.
- Attack vectors based on web framework.

* Your mileage may vary

Things to Check

Using Tools or Code Walk-throughs

- Passwords sent in a form without SSL
- SQL disclosure (in HTML comments, e.g.)
- Out of date web and application servers
- Web server and app information disclosure (in HTTP response headers, e.g.)
- Writable files and directories via HTTP, and directories with indexing turned on
- SQL error messages (another type of information disclosure)
- Form value caching (potential information leak)
- Password autocomplete enabled
- Forms submitted without using POST (forms not containing sensitive information are OK - PDS data set search form, for example)
- Cross-frame scripting
- Application exceptions displayed
- Open redirect (using URL parameters for redirect destinations)
- Methods other than GET and POST

Some tools are: Cenzic SmartAttacks, IBM Rational AppScan

Some site: https://www.owasp.org/

Refactor or Re-engineer for better security

- Compartmentalization of pdstools.arc.nasa.gov by physically separating from other Ames web applications.
- Running each server-side process as a unique user.
- Reduction of privileges of users to reduce possible damage in the event of a breach.
- Active monitoring of security advisories and prompt upgrades (US-CERT, Apache, etc.)
- Turn off or prohibit unnecessary features (using web and application server configuration, Java security profiles, etc.)

Specific Application Changes

- Verification that only stored queries are used (to avoid SQL injection)
- Verification that all input data is properly sanitized or HTML-escaped before display (to avoid XSS)
- Removal of HTML and Javascript comments (to avoid information leakage)
- Change cookie handling to "httpOnly", to help prevent XSS through cookies.
- Inhibit ability to run in a frame (to avoid social engineering attacks)
- Inhibit caching of form data in intermediate servers (to avoid information leakage)

Next Presentation

Physical layer security and other benefits of a reverse proxy architecture….

תודה
Dankie
Gracias
شكراً
Спасибо
Merci
Takk
Köszönjük
Terima kasih
Grazie
Dziękujemy
Dėkojame
Ďakujeme
Vielen Dank
Paldies
Kiitos
Täname teid
谢谢
Thank You
Tak
感謝您
Obrigado
Teşekkür Ederiz
감사합니다
Σας Ευχαριστούμ
ขอบคุณ
Bedankt
Děkujeme vám
ありがとうございます
Tack

PDS Planetary Data System

Scanning Tools

Open source:

- Burp suite - http://www.portswigger.net . Free and commercial tool. Excellent adjunct to manual testing and has a good scanner capability as well. Of professional web application testers I know, most use this.
- W3af - http://w3af.sourceforge.net/ - Open source scanning tool, seems to be developing quite a bit at the moment, primarily focuses on the automated scanning side of things, is still requires quite a bit of knowledge to use effectively.

Commercial tools available:

- Netsparker - http://www.mavitunasecurity.com/netsparker/
- IBM AppScan - http://www-01.ibm.com/software/awdtools/appscan/
- HP WebInspect - https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-201-200^9570_4000_100__
- Cenzic Hailstorm - http://www.cenzic.com/products/cenzic-hailstormPro/
- Acunetix WVS - http://www.acunetix.com/vulnerability-scanner/
- NTObjectives NTOSpider - http://www.ntobjectives.com/ntospider