Skipfish: web application security scanner

Brad Causey
OWASP Guy
IISFA Guy
brad.causey@owasp.org

OWASP Alabama
May, 20 2010

Presentation Overview

- What is Skipfish
- Why choose it
- Overview of features
- Demonstration
- My thoughts – Your thoughts

About Brad

- Web App Sec
- Forensics
- RO and Gun Enthusiast
- Father of 5

What is Skipfish?

- Web app sec reconnaissance tool
- Performs recursive crawl and dictionary-based probes
- Some security checks performed
- Robust reporting
- Credit goes to Michal Zalewski lcamtuf@google.com

Reconnaissance

- Uses Recursive Crawl, with boundaries
- Plain Text Dictionaries
- README-FIRST file
- Multiple to choose from
  - minimal.wl
  - extensions-only.wl
  - complete.wl
  - default.wl

Performance

- A major problem with many other scanners
- Typically between 500 and 7000 rps
- Forces response size restrictions (http/1.1 feature) to limit bandwidth saturation
- Utilizes response caching of items such as images/javascript
- Pure C, custom HTTP Stack optimized for speed

Features! (the good stuff!)

- Heuristic recognition of obscure paths/parameters (umm, really?)
- Identifies filters, path obfuscation, and frameworks
- Dynamic word list creation based on site
- Uses a “ratproxy” style passive scanner for csrf, MIME, and other directives
- Bundled security checks (new to me!)
- Post processing analysis

Detailed view of security checks

- High risk flaws (potentially leading to system compromise):
  - Server-side SQL injection (including blind vectors, numerical parameters).
  - Explicit SQL-like syntax in GET or POST parameters.
  - Server-side shell command injection (including blind vectors).
  - Server-side XML / XPath injection (including blind vectors).
  - Format string vulnerabilities.
  - Integer overflow vulnerabilities.
  - Locations accepting HTTP PUT.

Detailed view of security checks cont.

- Medium risk flaws (potentially leading to data compromise):
  - Stored and reflected XSS vectors in document body (minimal JS XSS support present) and HTTP redirects and HTTP header splitting.
  - Directory traversal
  - Assorted file POIs (server-side sources, configs)
  - Attacker-supplied script and CSS inclusion vectors (stored and reflected).
  - Mixed content problems on script and CSS resources (optional).
  - MIME types on renderables.
  - Incorrect or missing charsets on renderables.
  - Bad caching directives on cookie setting responses.

Detailed view of security checks cont.

- Low risk issues (limited impact or low specificity):
  - Directory listing bypass vectors.
  - Redirection to attacker-supplied URLs (stored and reflected).
  - Attacker-supplied embedded content (stored and reflected).
  - External untrusted embedded content.
  - Mixed content on non-scriptable subresources (optional).
  - HTTP credentials in URLs.
  - Expired or not-yet-valid SSL certificates.
  - HTML forms with no XSRF protection.
  - Self-signed SSL certificates.
  - SSL certificate host name mismatches.
  - Bad caching directives on less sensitive content.

Things to note

- Does not score well on the WASC criteria
  - Acknowledged by the author
  - Does it matter?
- Operating System Support
  - Linux
  - FreeBSD 7.0
  - MacOS X
  - Windows (Cygwin)

Things to note cont.

- Not the easiest tool to install
  - Libraries
  - Compilation
  - Dependencies:
    - GNU C Compiler
    - GNU Make
    - GNU C Library (including development headers)
    - zlib (including development headers)
    - OpenSSL (including development headers)
    - libidn (including development headers)
- No automatic dependency checking

Its Demo time!

Learn More

- http://code.google.com/p/skipfish/
- http://code.google.com/p/skipfish/wiki/SkipfishDoc
- http://code.google.com/p/skipfish/wiki/KnownIssues