Grendel-Scan

Distribution

- Written entirely in Java
- Uses Eclipse’s Standard Widget Toolkit (SWT)
- Windows, Linux and Mac builds
- Only requirement is JRE 1.5 or later

Primary Libraries

- Apache HTTP Components - http://hc.apache.org/
- Highly modified version of the Cobra HTML DOM parsing engine - http://lobobrowser.org/cobra.jsp
- Apache Derby (embeded SQL database) - http://db.apache.org/
- Mozilla Rhino (JavaScript engine) - http://www.mozilla.org/rhino/ Miscellaneous
- Apache Commons components - http://commons.apache.org/
- Nikto 2 database (used with permission)

Design Philosophy

- False positives vs. false negatives
  - False positives are easy to manually test for
  - False negatives require a full pen test to find
- Extensibility
  - Pushing abstract logic to shared libraries simplifies test module development

Application Walkthrough

Product Roadmap

- Version 1.1
  - Multi-part MIME encoded POST bodies
  - SSL/TLS configuration testing
  - PDF and XML report formats
  - Support for one-time passwords & authentication domains
  - Parameter incrementing
  - Upstream proxy authentication
  - Test module: Brute-force authentication
  - Test module: Error-based username enumeration

Product Roadmap

- Version 1.2
  - Automated AJAX navigation
  - Full featured HTTP fuzzer
  - Support for client SSL certificates
- Version 1.3
  - Reports of new and remediated vulnerabilities between scans
  - Support for graphs in reports
  - Ability to save and resume scans

Demonstration Environment

- SLAX-based LiveCD
- Server (Typical LAMP Stack):
  - Apache HTTPD (from Slackware, defaults + mod_php)
  - MySQL (from Slackware, defaults)
  - PHP 4
  - Zencart 1.1.2 (c. February 2004, known vulnerabilities)
- Client
  - Mozilla FireFox 3.0
  - Grendel-Scan

Grendel-Scan Demonstration: Automated & Manual Testing

Advantages of Automated Web Scanners

- Minimal training requirements
- Fast
- Cheap

Limitations of Automated Web Scanners

- Automated scanners cannot generally detect:
  - Logic flaws (e.g. send -$1000 to another account)
  - Design flaws (e.g. weak password recovery questions)
  - Improper application flow enforcement (e.g. forced browsing)
- Other limitations
  - Scanners cannot contextually understand an application’s logic or data
  - Scanners typically generate far more traffic than manual tests