A SAMURAI-WTF INTRO TO THE ZED ATTACK PROXY

Justin Searle – justin@utilisec.com - @meeas
Samurai-WTF

• 2 Versions: Live DVD and VMware Image
• Based on Ubuntu Linux
• Over 100 tools, extensions, and scripts, included:
  – w3af
  – BeEF
  – Burp Suite
  – Grendel-Scan
  – DirBuster
  – Maltego CE
  – Nikto
  – WebScarab
  – Rat Proxy
  – nmap
Project URLs

- **Main project page:**
  - [http://www.samurai-wtf.com](http://www.samurai-wtf.com)

- **Support information (and tracker) at:**
  - [http://sourceforge.net/projects/samurai/support](http://sourceforge.net/projects/samurai/support)

- **Development mailing list at:**
  - [https://lists.sourceforge.net/lists/listinfo/samurai-devel](https://lists.sourceforge.net/lists/listinfo/samurai-devel)

- **SVN repository:**
  - `svn co https://samurai.svn.sourceforge.net/svnroot/samurai samurai`

- **Project Leads:**
  - Kevin Johnson - kjohnson@secureideas.net - @secureideas
  - Justin Searle - justin@utilisec.com - @meeas
  - Frank DiMaggio - fdimaggio@secureideas.net - @hanovrfst
  - Raul Siles - raul@taddong.com - @taddong
Logging In

• Username: samurai
• Password: samurai
• http://whatisthesamuraipassword.com
Formal Methodology

• A simple methodology:
  – Recon: Before touching the app
  – Mapping: Learning the app from a user/developer's perspective
  – Discovery: Learning the app from an attacker's perspective
  – Exploitation: Need I say more?!?

• Every step leads to new insight into the application and target environment
• New insight provides additional opportunities for previous phases
Methodology in Real Life

- We still follow the overall clockwise flow, but we often move back and forth between processes.
- Trick is to keep focused and progressing through the steps.
- General rule for deviation from clockwise progression:
  - 5 attempts or 5 minutes.
Zed Attack Proxy (ZAP)

- **Lead:** Simon Bennetts (Psiinon)
- **Site:** code.google.com/p/zaproxy
- **Purpose:** An interception proxy with integrated tools to find vulnerabilities. This project was forked from Paros Proxy and is actively maintained (unlike Paros).
- **Language:** Java
- **Notable Features:**
  - Port Scanner
  - Automated and Passive Scanner
  - Spider, Brute Force, Fuzzing tools
  - Adding Notes and Alerts to request/response pairs
  - Great "Filters" which allow logging of unique elements and auto regex search/replace
Updating ZAP

• Simon and team did a special ZAP 1.3.4 release for this Black Hat workshop to provide us the following new features:
  – Custom input files for the Fuzzing and Brute Force tools
  – Ability to disable recursion in the Brute Force tool
  – Inverse regex searches and Fuzz match highlighting
  – Support for cookies and POST data in third party tools

• We'll be using this version (1.3.4) for this workshop
  – Download it at:  http://code.google.com/p/zapproxy
  – Extract the files and run "zap.sh"
ZAP's Extra Polish

- Beautiful Java UI regardless of OS
- Built in user documentation and help pages
- Automatically checks for updates
- Flexible UI allows you to focus on important items
- Universal status bar for all tools in one place
- Supports 11 languages and growing
- REST API for advanced users (http://zap)
Using Firefox with ZAP

• Configuring Firefox to trust ZAP's dynamic SSL certificates
  – Have ZAP generate a SSL Root CA
  – Save the certificate to your file system
  – Import it into FireFox

• Use Foxy Proxy to quickly configure Firefox to use ZAP as a proxy
TODAY'S TARGET: DVWA

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a classroom environment.
Damn Vulnerable Web App (DVWA)

- Project Lead: Ryan Dewhusrt (ethicalhack3r)
- Site: http://sourceforge.net/projects/dvwa
- Purpose: a light weight PHP/MySQL web application that is easy to use and full of vulnerabilities to exploit. Used to learn or teach the art of web application security
- Language: PHP
- Accessing:
  - http://dvwa
  - https://dvwa
    - admin
    - password
- Notable features:
  - GET requests
  - 3 difficulty levels
  - Includes PHP IDS
Demo Tasks: DVWA Mapping

1. Port Scanning in ZAP
2. Basics techniques to manual map an application
   - Does the application authenticate users?
     - How do you login and logout?
     - How does the application track session state?
     - How do update account settings such as passwords?
     - How do you reset or recover an account?
   - Where does the application accept user input?
     - Which inputs are reflected back to the user?
     - Which inputs might be used in queries to a database?
     - Which inputs might be used in system tools or file names?
   - Which pages return the slowest or fastest?
   - Which pages are dangerous for automated tools?
3. Adding alerts for manual findings
4. Using the Spider tool to finish mapping DVWA
Demo Tasks: DVWA Discovery

1. Finding unlinked resources with the Brute Force tool
2. Passive vulnerability scans
3. Active vulnerability scans
4. Third party tool integration
   - We'll be using nikto for the demo
   - Syntax:

```
nikto -host %site% -port %port%
```

1. Fuzzing
Demo Tasks: DVWA Exploitation

- Leveraging the Fuzzing tool to enumerate commands
- Using third party tools inside ZAP
  - We'll be using sqlmap for this today
  - Syntax:

  sqlmap -u %url% --cookie %cookie% -v 0 --drop-set-cookie --dbs
Contact Information

Upcoming SamuraiWTF Black Hat courses:
- Amsterdam – Mar. 14, 2012 (one-day version)

Justin Searle
Managing Partner – UtiliSec
justin@utilisec.com
801-784-2052
@meeas

Kevin Johnson
Owner – SecureIdeas
kjohnson@secureideas.net
904-639-6709
@secureideas