



# Impact of Plugins on Web Application Security

Cincinnati Chapter Meeting  
June 29<sup>th</sup>, 2010  
James Walden and Maureen Doyle  
Northern Kentucky University

**OWASP**

Copyright © The OWASP Foundation  
Permission is granted to copy, distribute and/or modify this document  
under the terms of the OWASP License.

**The OWASP Foundation**  
<http://www.owasp.org>

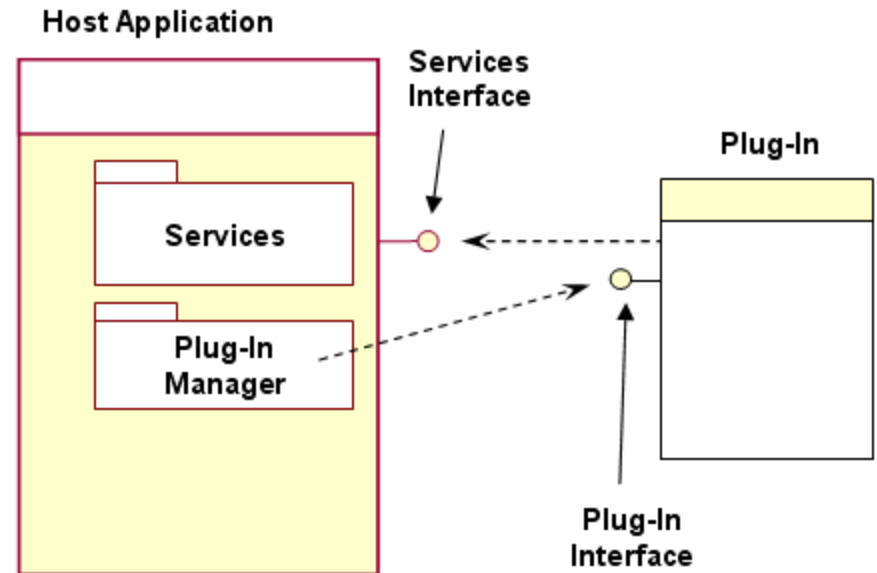
# Topics

1. Plugins
2. Measuring Vulnerabilities
3. Plugin Vulnerabilities
4. Comparing Core and Plugin Security
5. OWASP Top 10 Vulnerabilities
6. Conclusions

# Plugins

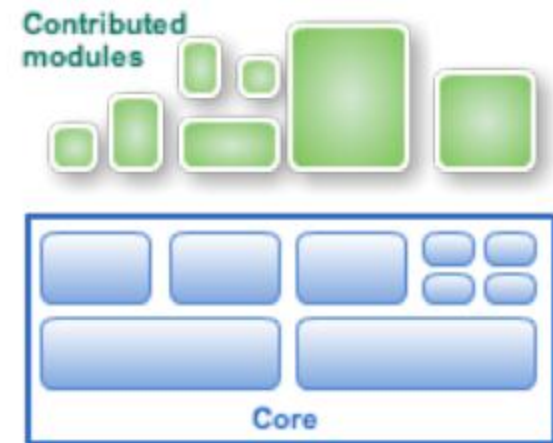
Plugins add features to web applications:

- ▶ Advertising
- ▶ E-commerce
- ▶ Media
- ▶ Security
- ▶ Site Navigation
- ▶ Statistics
- ▶ Themes
- ▶ User Management



# What makes up a web application?

- Is it the core code or code code + plugins?
  - ▶ Some apps are almost always deployed with plugins.
  - ▶ Plugins are written by non-core developers.
  - ▶ Core site may or may not track plugin security.
- Some apps are packaged in distributions with plugins such as Drupal which has:
  - ▶ OpenAtrium (Development Seed)
  - ▶ Acquia Drupal
  - ▶ OpenPublish
  - ▶ Pressflow (Four Kitchens)



# Research Objective

**Goal:** Identify differences between security of core code and plugins for web applications.

## Research questions:

1. Are plugins less secure than core code?
2. How are vulnerabilities distributed across plugins?
3. How do different applications compare in terms of plugin security?

# Measuring Vulnerabilities

## Reported Vulnerabilities in NVD or OSVDB

- ▶ Coarse-grained time evolution.
- ▶ Difficult to correlate with revision.
- ▶ Undercounts actual vulnerabilities.

## Dynamic Analysis

- ▶ Expensive.
- ▶ False positives and negatives.
- ▶ Requires installation of application.

## Static Analysis

- ▶ False positives and negatives.
- ▶ Static Analysis Vulnerability Density =  $\text{vulns/kloc}$ .

# Measuring Web Application Vulnerabilities

- NVD doesn't offer a web application category.
- Even if they did
  - ▶ Commercial web sites don't require users to patch, so vulnerabilities are rarely sent to public vuln DBs.
  - ▶ We have to report on open source vulnerabilities.
- Advantages of open source
  - ▶ Publicly reported vulnerabilities.
  - ▶ Source code available to measure vulnerabilities.
  - ▶ Source code available for software metrics.
  - ▶ Multiple versions of source code available, making it possible to do time comparisons.

# Open Source Web Applications

## Selection process

- ▶ PHP web applications from freshmeat.net.
- ▶ A central plugin repository.
- ▶ Automatable downloads.
- ▶ At least 10 plugins.

## Why PHP?

- ▶ Most popular web applications written in PHP.
- ▶ Can compare applications evenly.

## Range of projects

- ▶ 12 projects met selection criteria.
- ▶ 13,535 plugins for these applications.
- ▶ Plugins per app ranged from 10 to 8989 plugins.



# Open Source Applications are Targets

The screenshot shows a Mozilla Firefox browser window displaying the WPSecurityLock blog. The browser's address bar shows the URL <http://www.wpssecuritylock.com/blog/>. The page features a dark header with the WPSecurityLock logo and navigation links: HOME, BLOG, SERVICES, PRODUCTS, EVENTS, JOIN, TESTIMONIALS, ABOUT, CONTACT. A search bar is also present.

The main content area is titled "Blog" and contains three articles:

- WordPress 3.0 Thelonious Security Features**: After much anticipation, the stable version of WordPress 3.0 (Thelonious) was released on Thursday, June 17, 2010. According to Matt Mullenweg, WordPress 3.0 is faster, stabler and more secure. It comes...  
Tags: [thelonious](#), [upgrade wordpress](#), [wordpress 3.0](#), [wordpress security](#) 22 Comments
- Breaking News: WordPress Hacked with cloudisthebestnow on Go Daddy**: On June 8, 2010 at approximately 3pm EST self-hosted WordPress blogs, along with other PHP based websites started getting attacked with cloudisthebestnow malware. This is a server-side hacker attack. We...  
Tags: [base64\\_decode](#), [cloudisthebestnow](#), [Go Daddy hacked again](#), [godaddy hacked](#), [how to fix hacked wordpress](#), [wordpress hacked](#) 20 Comments
- Breaking News: WordPress Hacked with losotrana on Go Daddy**: Reports of WordPress blogs self-hosted at GoDaddy.com and have been infected with the losotrana[dot]com/js.php on Monday, May 17, 2010 and Thursday, May 20, 2010. Warning: This is dangerous malware! This...  
Tags: [base64\\_decode](#), [how to fix hacked wordpress](#), [losotrana.com/js.php](#), [mediatemple hacked](#), [webguardyourpc hack](#), [wordpress hacked](#) 85 Comments

The right sidebar contains a section titled "Secure Your WordPress Now" with a form to subscribe to WPSecurityLock News. Below this is a "Latest Blog Posts" section listing the same three articles. At the bottom of the sidebar are social media links for Facebook and Twitter.



# Results

Plugins slightly less secure than core.

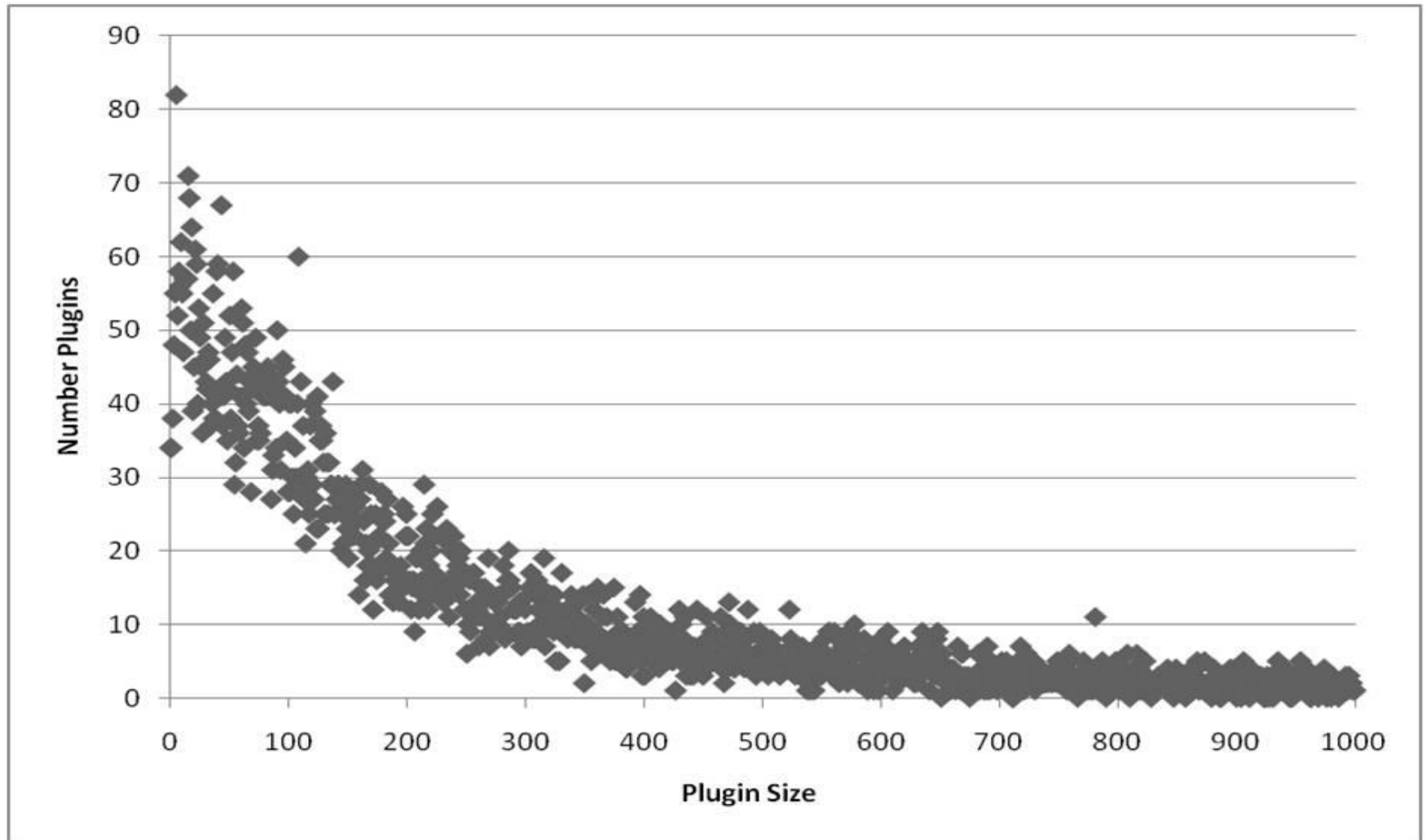
- ▶ Plugins made up 91% of 11.7 MLOC.
- ▶ Contained 92% of 135,907 vulnerabilities.

Plugin SAVD correlates strongly with code size.

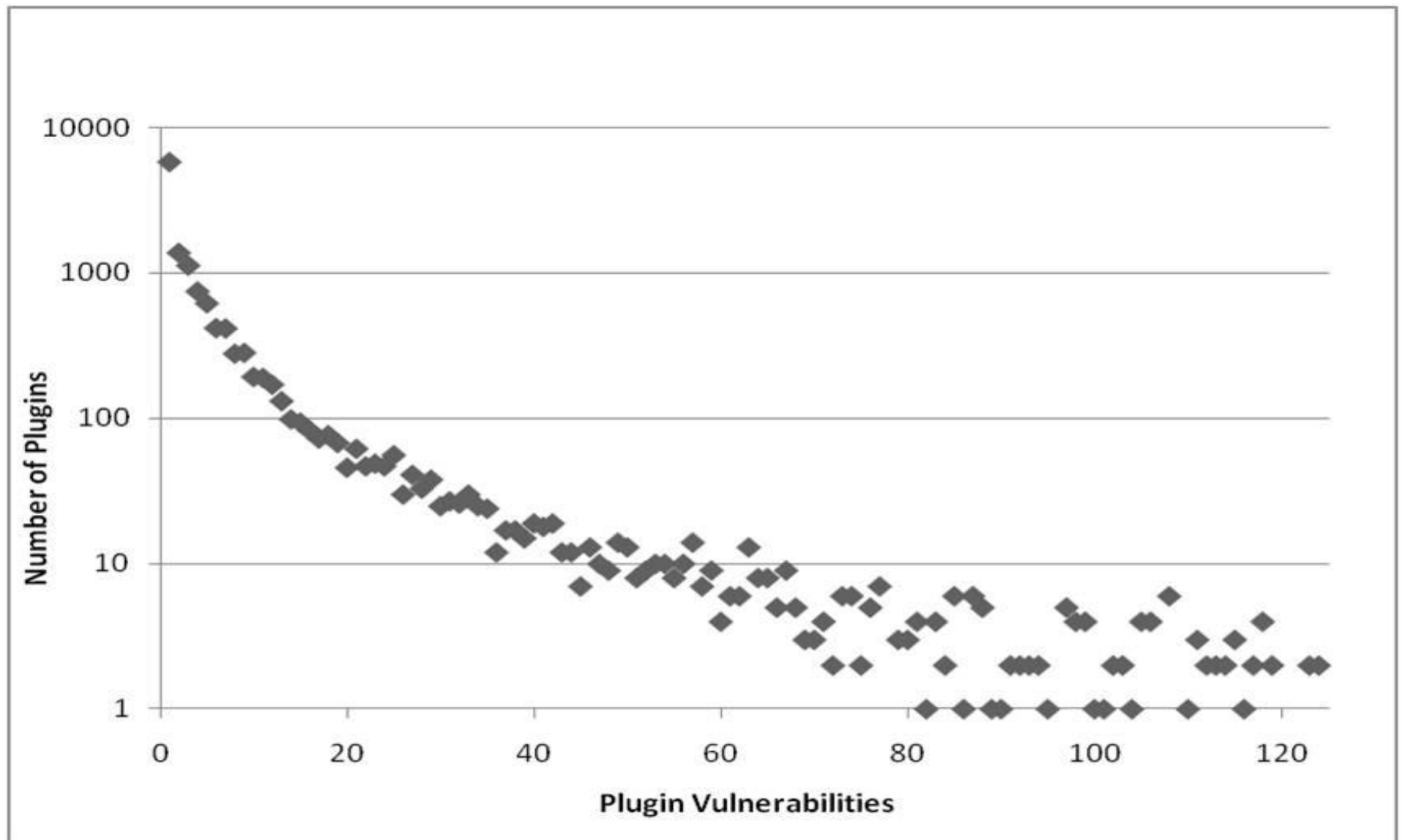
- ▶  $\rho = 0.91$ .
- ▶ Larger plugins are more likely to have vulnerabilities.

Core SAVD does not correlate with code size.

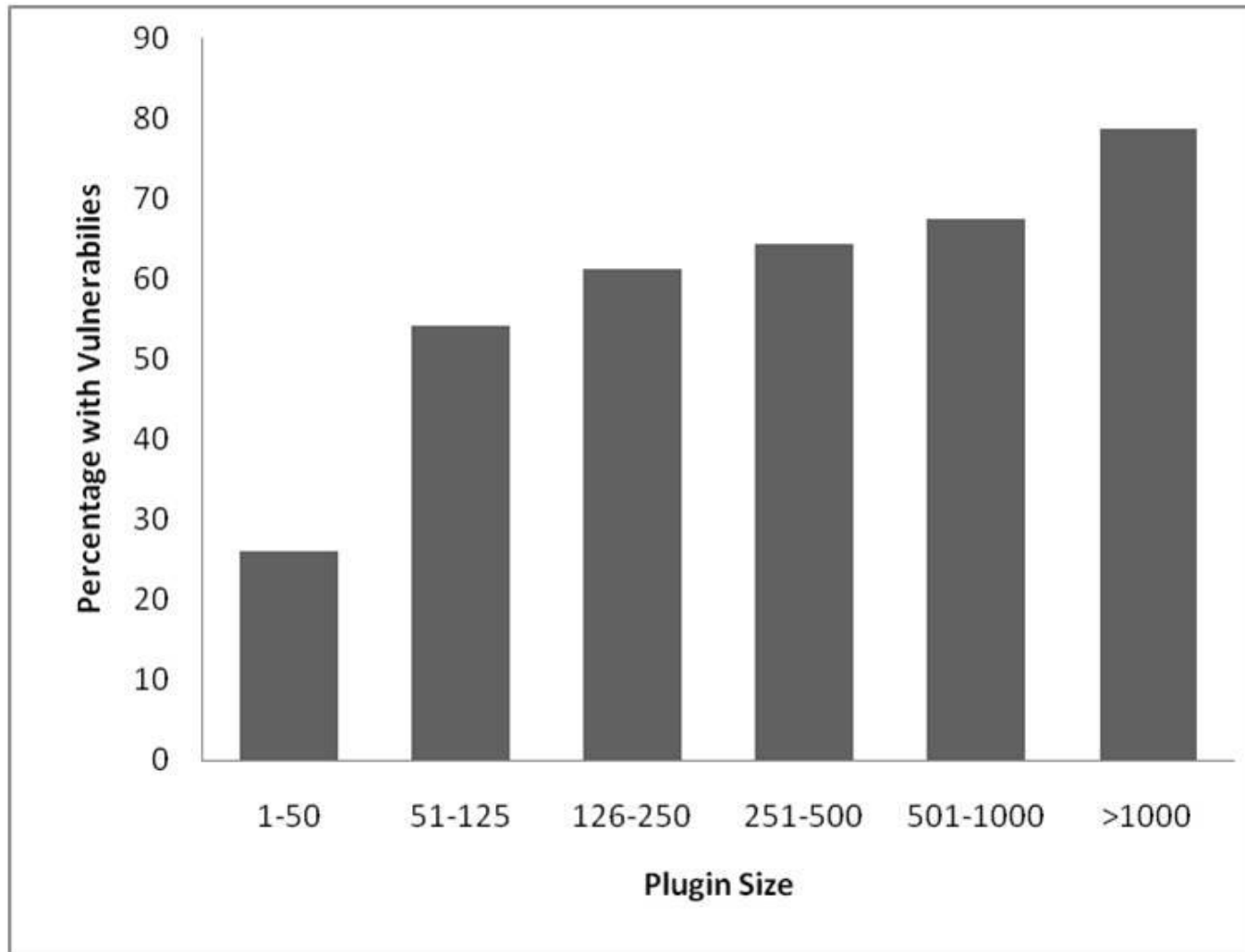
# Plugin Size Distribution



# Plugin Vulnerability Distribution



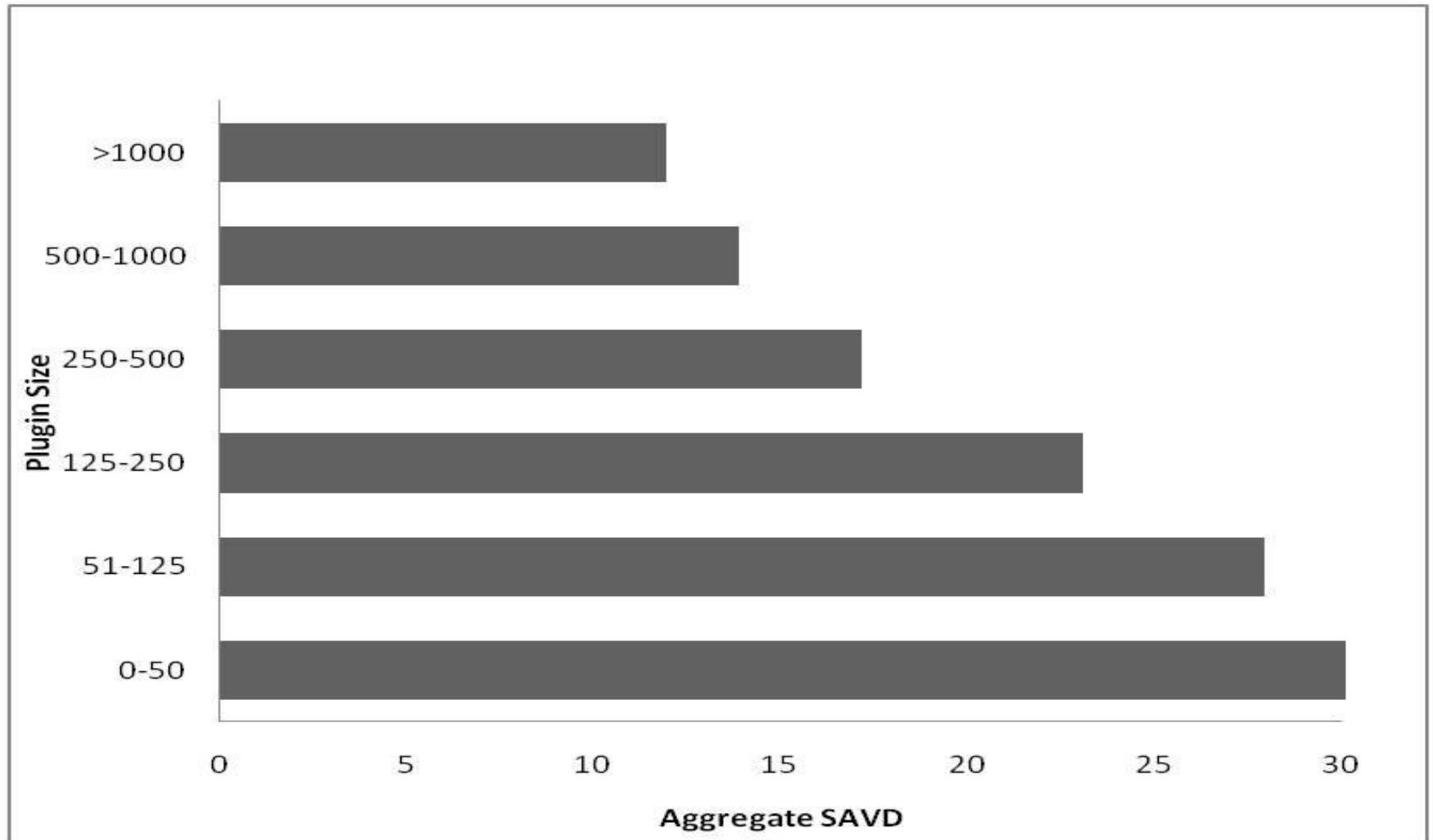
# Percentage of Vulnerable Plugins by Size



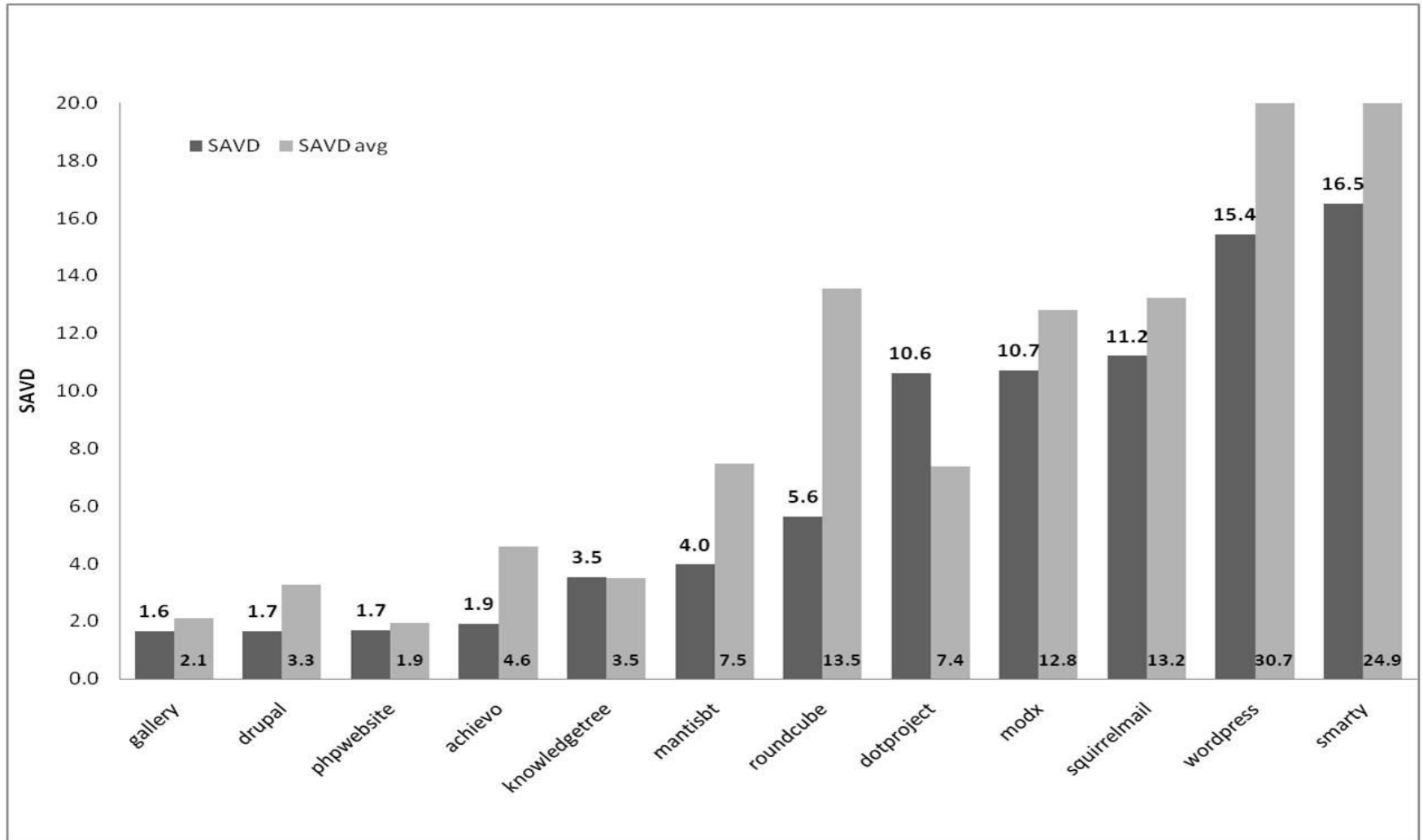
# Static Analysis Vulnerability Density (SAVD)

- Number of vulnerabilities found by a static analysis tool per 1000 lines of source code.
  - ▶ Fortify SourceAnalyzer 5.8.0
- Aggregate SAVD
  - ▶ Use aggregate of source code for all plugins.
  - ▶  $\text{Total vulnerabilities} / \text{Total KSLOC}$
- Average SAVD
  - ▶ Compute SAVD for each plugin individually.
  - ▶ Average individual plugin SAVD values.

# SAVD by Plugin Size

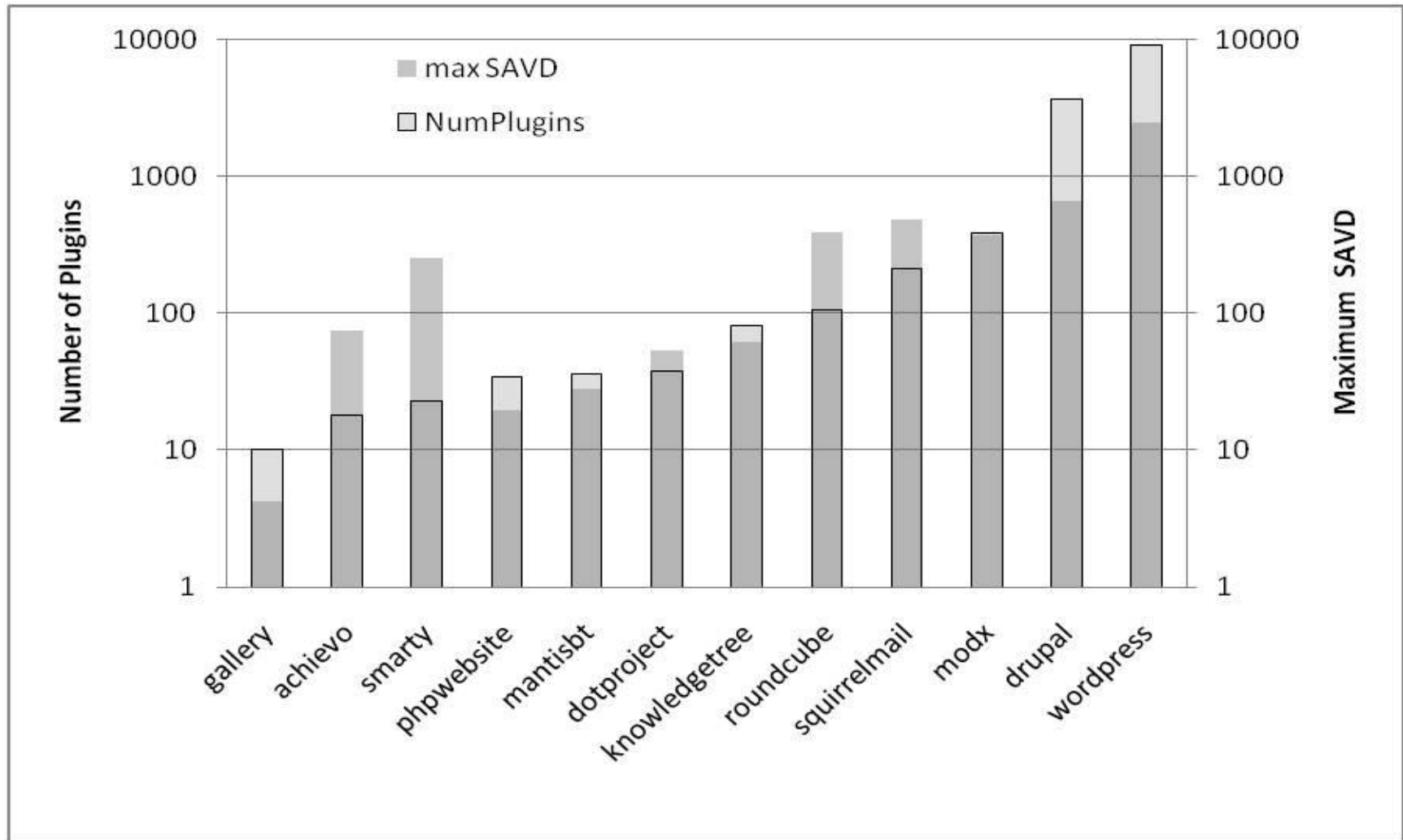


# Average vs. Aggregate SAVD of Plugins





# Plugin Counts and Maximum Plugin SAVD



# Do plugins make your site less secure?

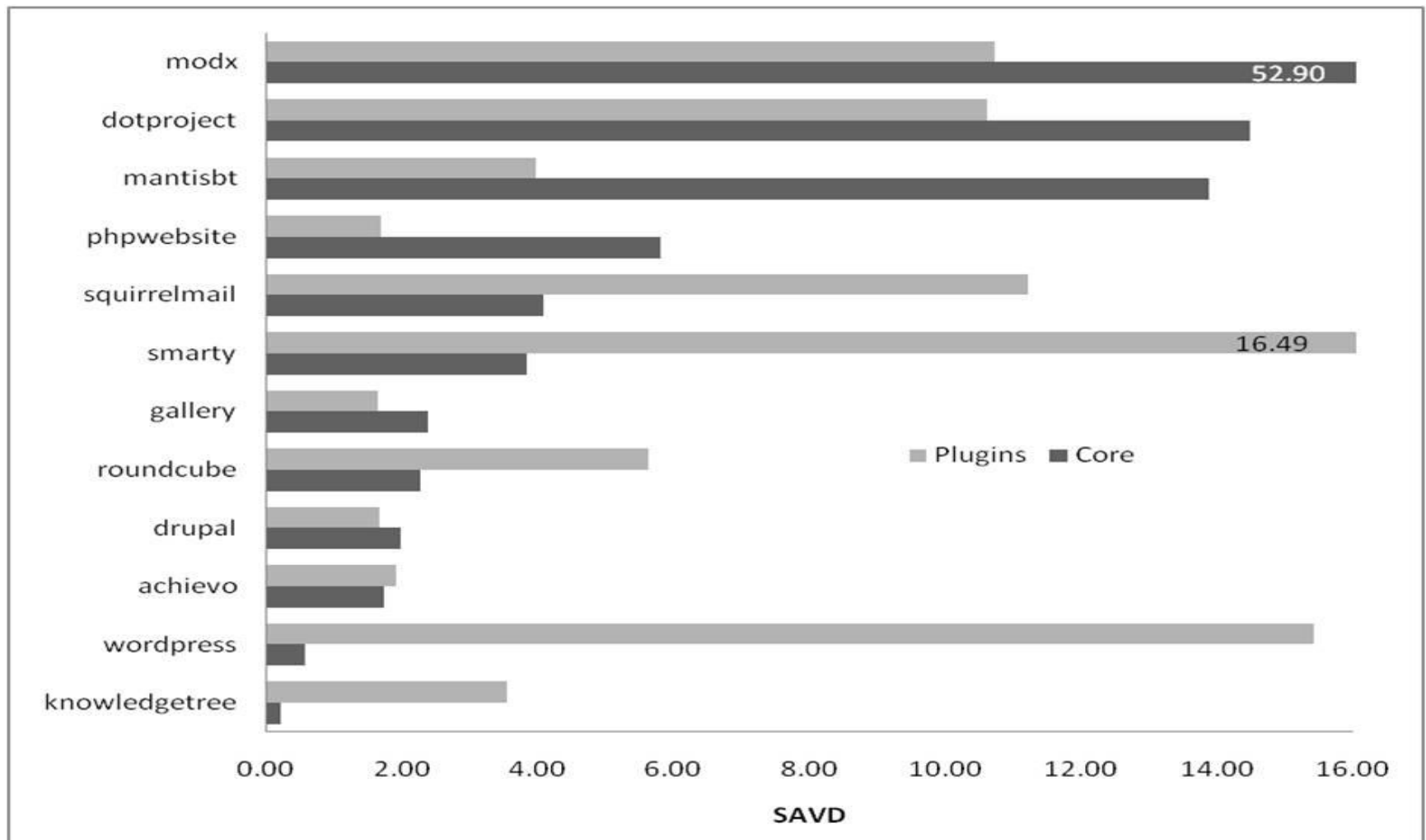
Core code developed by small core team.

- ▶ Team experienced with core code over years.
- ▶ May or may not be paid full-time developers.
- ▶ Most sites have some form of security information.

Plugins developed by many people.

- ▶ Wide variety of programming experience.
- ▶ Few develop more than one plugin and so have little experience with application compared to core team.
- ▶ Few plugins mention security unless a vulnerability has been previously reported.

# Core vs. Plugin SAVD



# Drupal Core vs. Plugins

- Drupal tracked both core and plugin vulns since 2006.

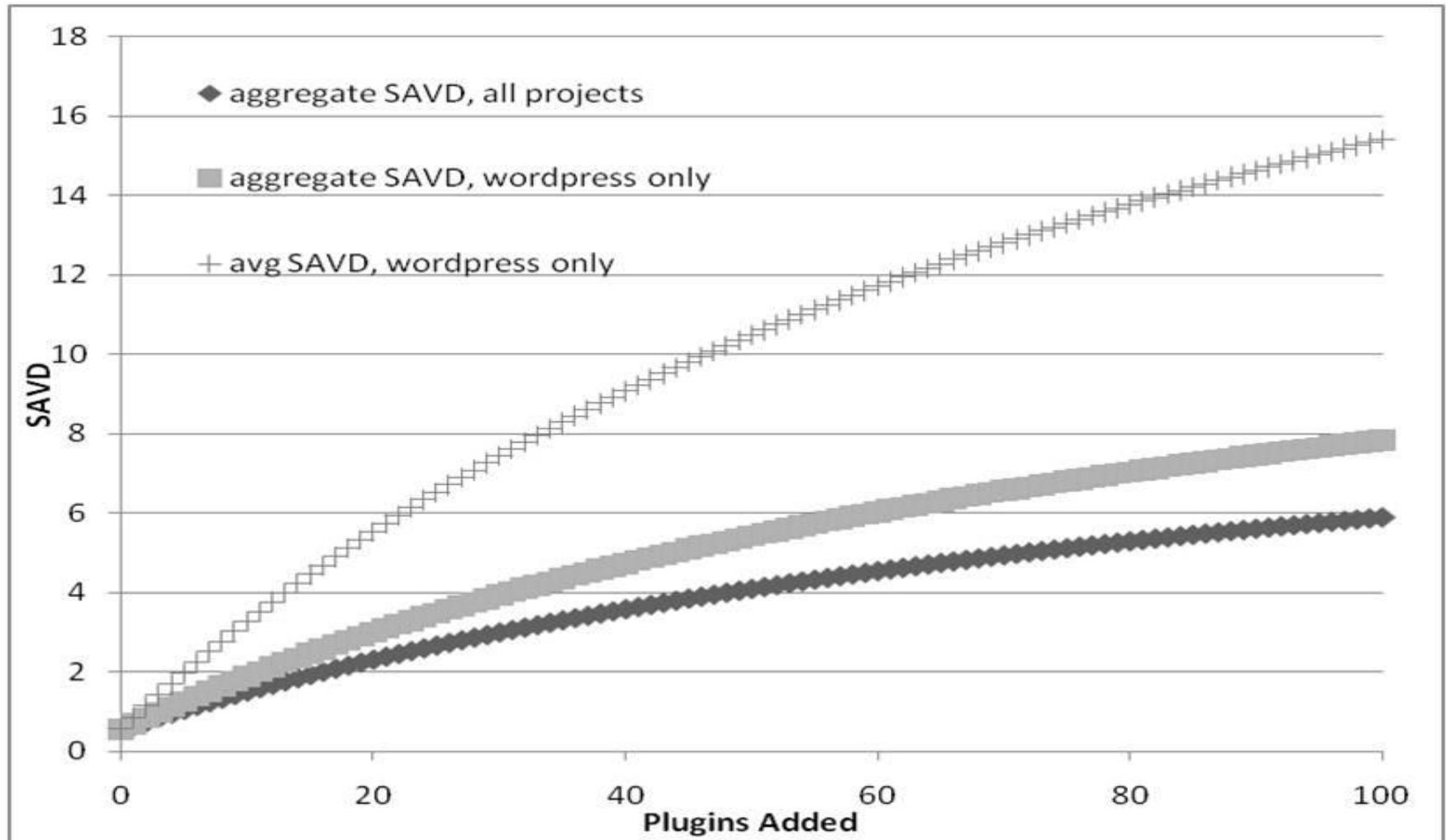
Year	Core	Contrib
2009	8	115
2008	11	64
2007	10	22
2006	12	21

- Most popular CMS with 1.58% of web sites including whitehouse.gov

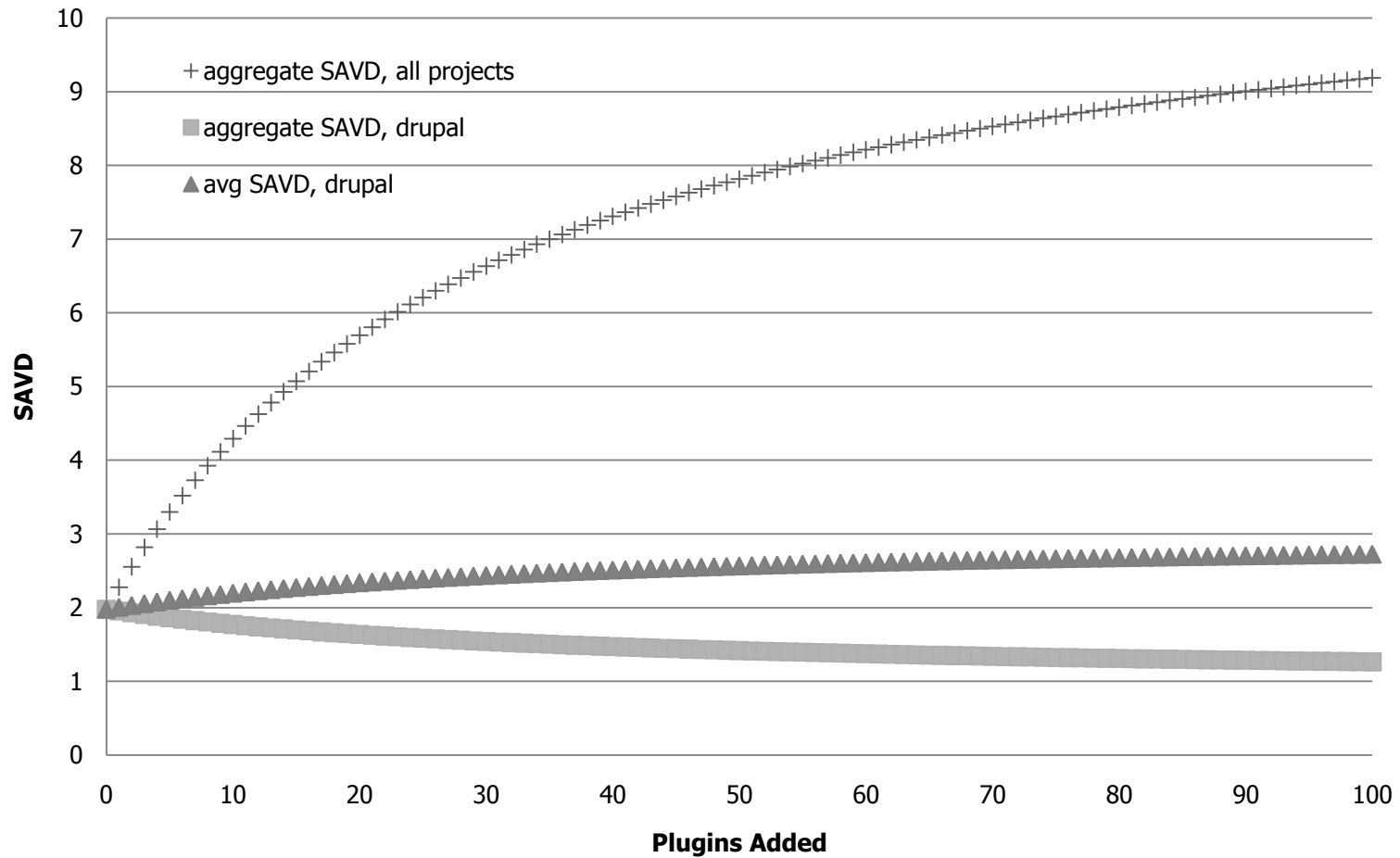
[www.drupalsecurityreport.org](http://www.drupalsecurityreport.org)

- Secure coding documentation.
- XSS Filter API.
- DB API to handle SQLi attacks.
- Input validation API.

# WordPress: Effect of Adding Plugins on SAVD



# Drupal: Effect of Adding Plugins on SAVD



# Vulnerability Categories

## Mapped Fortify categories to OWASP Top 10 2010.

- ▶ SCA 5.8 reports 73 categories, only 25 in this code.
- ▶ 18 of 25 categories mapped to 5 of OWASP Top 10.
- ▶ 7 remaining categories did not map to Top 10.

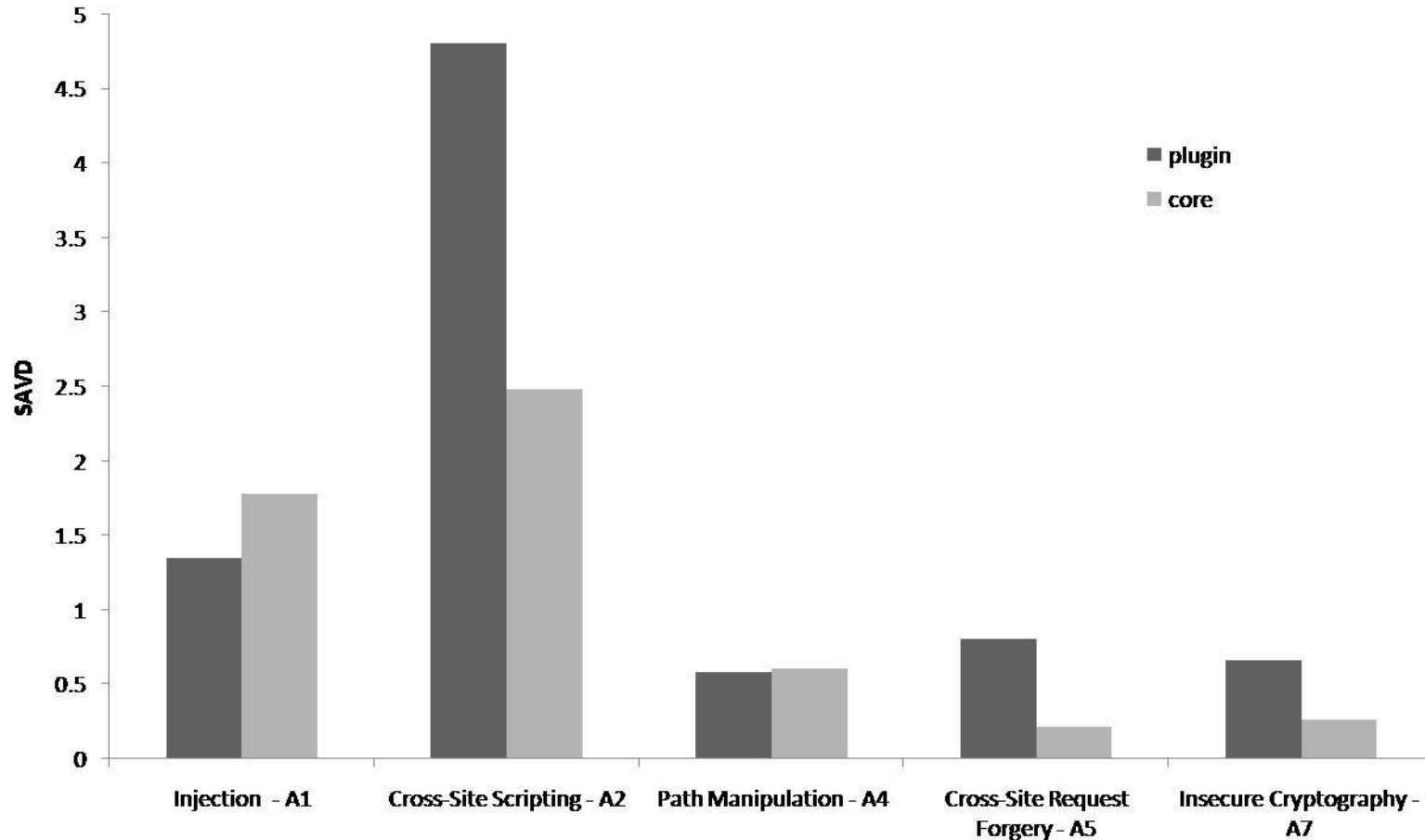
### T10

### OWASP Top 10 Application Security Risks – 2010

- ✓ **A1 – Injection**
  - Injection flaws, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data.
- ✓ **A2 – Cross-Site Scripting (XSS)**
  - XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation and escaping. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
- A3 – Broken Authentication and Session Management**
  - Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, session tokens, or exploit other implementation flaws to assume other users' identities.
- ✓ **A4 – Insecure Direct Object References**
  - A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.
- ✓ **A5 – Cross-Site Request Forgery (CSRF)**
  - A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.
- A6 – Security Misconfiguration**
  - Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. All these settings should be defined, implemented, and maintained as many are not shipped with secure defaults. This includes keeping all software up to date, including all code libraries used by the application.
- ✓ **A7 – Insecure Cryptographic Storage**
  - Many web applications do not properly protect sensitive data, such as credit cards, SSNs, and authentication credentials, with appropriate encryption or hashing. Attackers may steal or modify such weakly protected data to conduct identity theft, credit card fraud, or other crimes.
- A8 – Failure to Restrict URL Access**
  - Many web applications check URL access rights before rendering protected links and buttons. However, applications need to perform similar access control checks each time these pages are accessed, or attackers will be able to forge URLs to access these hidden pages anyway.
- A9 – Insufficient Transport Layer Protection**
  - Applications frequently fail to authenticate, encrypt, and protect the confidentiality and integrity of sensitive network traffic. When they do, they sometimes support weak algorithms, use expired or invalid certificates, or do not use them correctly.
- A10 – Unvalidated Redirects and Forwards**
  - Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

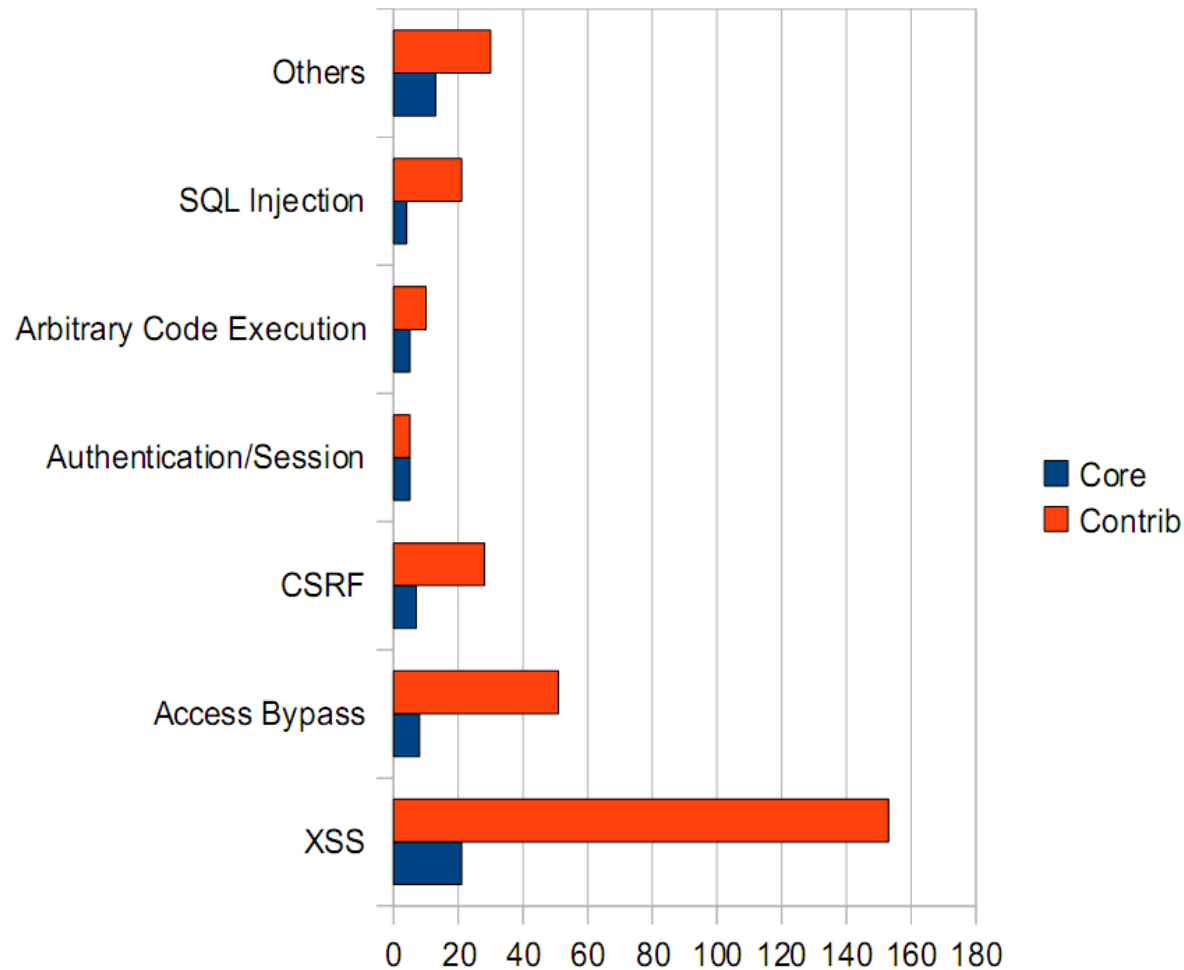


# OWASP Top 10: Core vs. Plugin SAVD

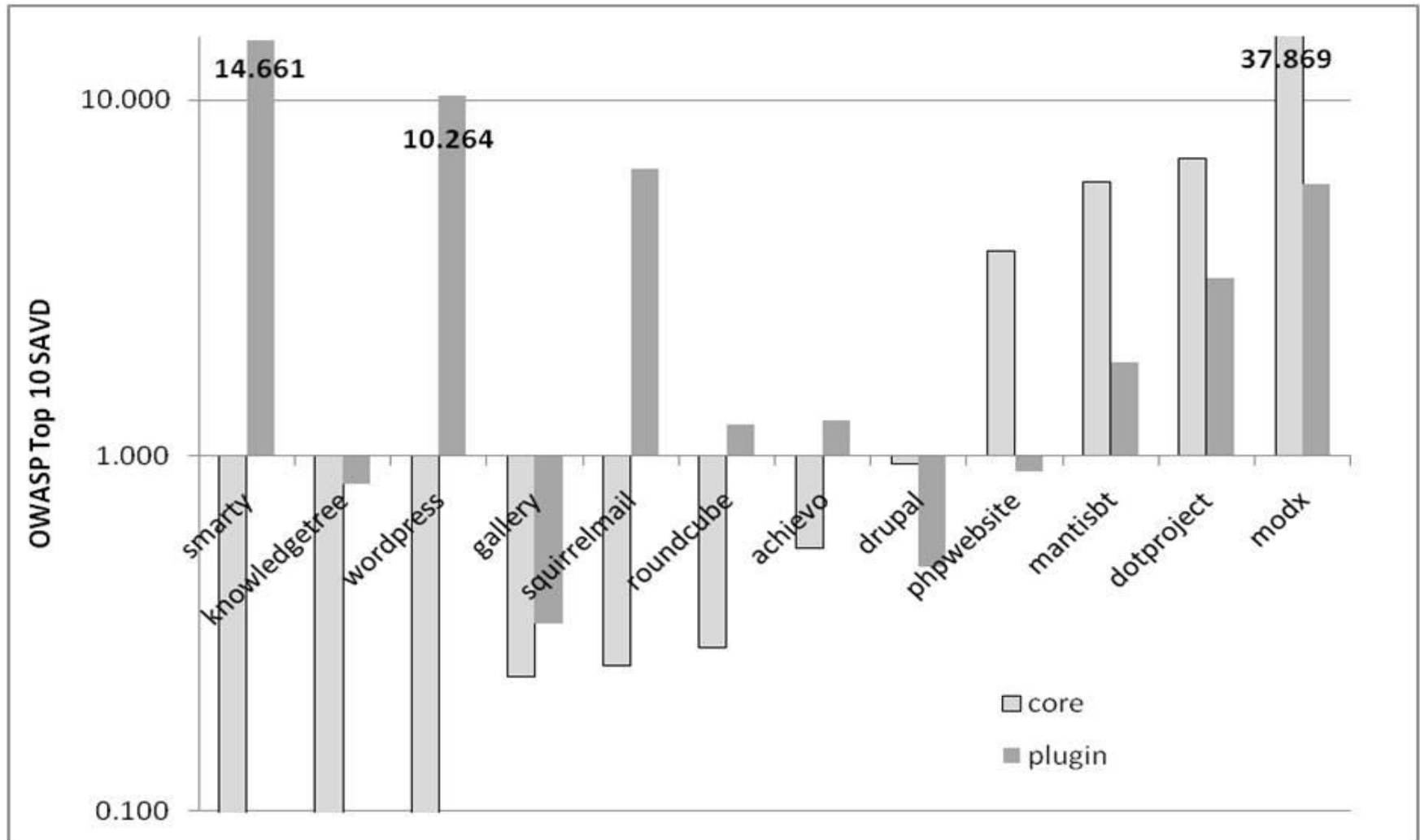




# Drupal: Core vs. Plugins by Category



# OWASP Vulnerabilities: Core vs. Plugin by App



# Conclusions

- Plugin code is not always worse than core code.
  - ▶ Older apps with more plugins tend to have more secure core code.
  - ▶ Security documentation tends to indicate apps with more secure core code.
  - ▶ Large number of NVD vulnerabilities does not necessarily indicate poor security.
- Plugin size is important for security
  - ▶ 30% of plugins <50 lines have vulnerabilities
  - ▶ Over 50% of plugins >50 lines have vulnerabilities