

Building Secure Applications:

Avoiding the SANS Top 25 Most Dangerous Programming Errors

Chris Wysopal
Veracode CTO

VERACODE

Top causes of computer intrusions

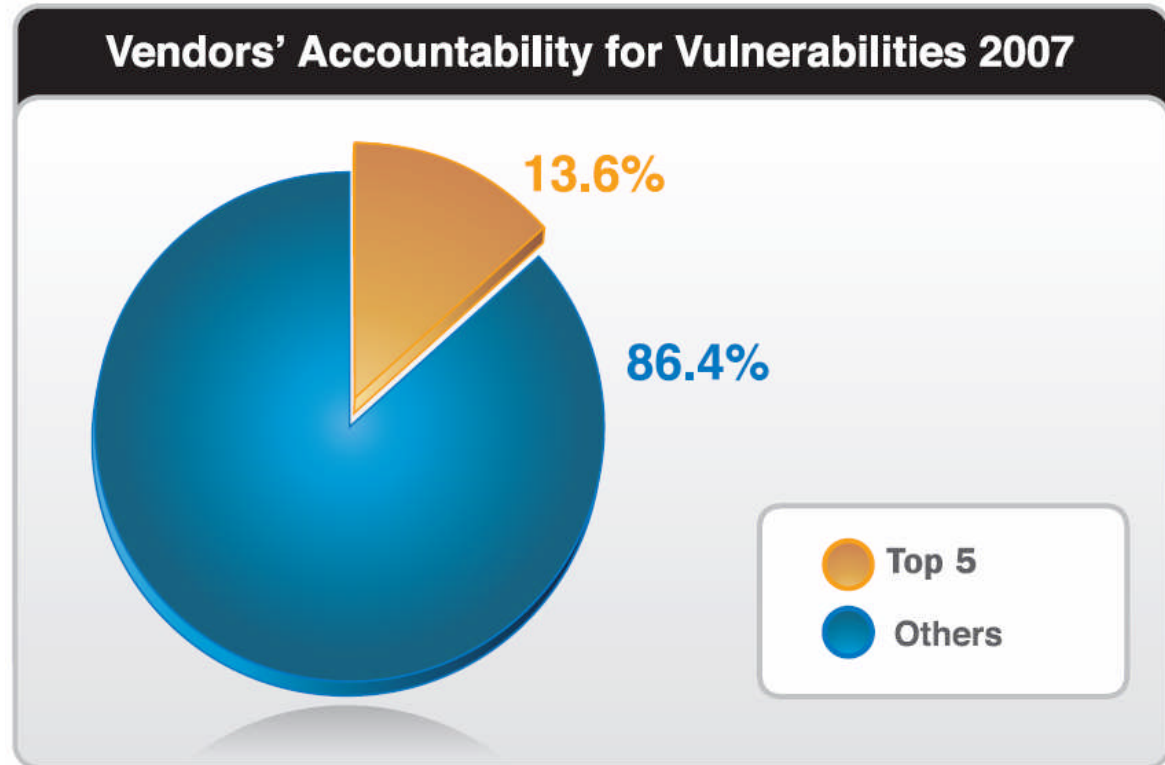
1. **Vulnerability in underlying OS or other “infrastructure” software**
 - Steady stream of updates from Microsoft, Apple, Oracle
2. **Misconfiguration of network or host**
 - weak/no passwords, world readable file shares
3. **“Social Engineering” – tricking the user**
 - Download and run malicious codec or eCard, phishing
4. **Vulnerability in applications**
 - Media players, desktop software, web applications

**SANS/CWE Top 25 Programming Errors
addresses #1 and #4**

Myth – All Vulnerabilities are from Large Software Vendors

Vendor	Vulnerabilities Reported in 2007
Microsoft	238
Apple	207
Oracle	183
IBM	137
Cisco	113

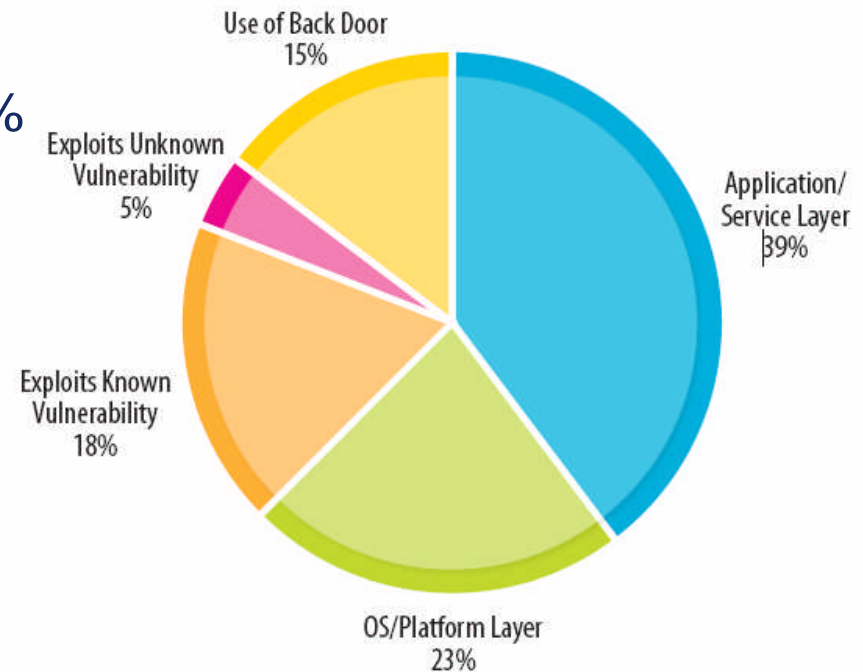
Source: IBM X-Force 2007 Security Trends Report



“Microsoft executives said they were pleased with the progress made since the company was shaken by a series of destructive programs that spread rapidly around the world over the Internet beginning in 2003. But they said that unless software development practices change throughout the industry, any improvements in the security of Windows would be meaningless.” – New York Times, Nov 3, 2008

Verizon Data Breach Report

- **Four years of forensic research covering 500 cases**
- **59% of breaches involved hacking**
 - Application/Service layer – 39%
 - OS/Platform layer – 23%
 - Exploit known vulnerability – 18%
 - Exploit unknown vulnerability – 5%
 - Use of backdoor – 15%



Recent Top 25 Real-World Examples

▪ February 6, 2009

- SQL Injection vulnerability in Kaspersky support web site
- `http://usa.kaspersky.com/support/208279383000+UniOn+aLL+SELECT+1,COLUMN_name,3,4+FROM+INFORMATION_schema.columns+WHERE+TABLE_name=CHAR(117,115,101,114,115)--/`



▪ March 26, 2008

- Private photos exposed in Facebook due to insufficient authorization
- Similar issues in third-party Facebook apps





Common Weakness Enumeration
A Community-Developed Dictionary of Software Weakness Types

- **Targeted to developers and security practitioners, the Common Weakness Enumeration (CWE) is a formal list of software weakness types created to:**
 - Serve as a common language for describing software security weaknesses in architecture, design, or code.
 - Serve as a standard measuring stick for software security tools targeting these weaknesses.
 - Provide a common baseline standard for weakness identification, mitigation, and prevention efforts.
- **Over 700 flaw types.**

CWE-121: Stack-based Buffer Overflow

Weakness ID: 121 (*Weakness Variant*)

Description

- A stack-based buffer overflow condition is a condition where the buffer being overwritten is allocated on the stack (i.e., is a local variable or, rarely, a parameter to a function).

Alternate Terms

- Stack Overflow
- "Stack Overflow" is often used to mean the same thing as stack-based buffer overflow, however it is also used on occasion to mean stack exhaustion, usually a result from an excessively recursive function call. Due to the ambiguity of the term, use of stack overflow to describe either circumstance is discouraged.

Common Consequences

- Availability - Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop.
- Access Control - Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy.
- Other - When the consequence is arbitrary code execution, this can often be used to subvert any other security service.

Likelihood of Exploit

- Very High

SANS/CWE Top 25 Programming Errors



- **List was selected by a group security experts from 34 organizations including:**
 - Academia: Purdue, Univ. of Cal., N. Kentucky Univ.
 - Government: CERT, NSA, DHS
 - Software Vendors: Microsoft, Oracle, Red Hat, Apple
 - Security Vendors: Veracode, Fortify, Cigital, Symantec

- **List is made up of the root causes of vulnerabilities and not a list of attacks**
 - SQL Injection is the attack. Improper Input validation and Failure to Preserve SQL Query Structure are the errors.

- **Selection criteria was the errors that led to serious breaches**
 - prevalence of error combined with seriousness of impact

Prevalence based on 2008 CVE data

Category	Count	%
SQL Injection	941	19.4%
XSS	681	14.0%
Buffer Overflow	455	9.4%
Directory Traversal	298	6.1%
PHP Include	135	2.8%
Symbolic Link	133	2.7%
Authorization Bypass	113	2.3%
DoS Malformed Input	97	2.0%
Information Leak	84	1.7%
Integer Overflow	78	1.6%
CSRF	57	1.2%
Bad Permissions	40	0.8%
Unnecessary Privileges	36	0.7%

Hard coded Password	36	0.7%
Upload of code	34	0.7%
Weak Crypto	30	0.6%
Format String	26	0.5%
Insufficient Randomness	24	0.5%
Metacharacter Injection	23	0.5%
Search Path	20	0.4%
Memory Leak	18	0.4%
Sensitive data root	16	0.3%
Race Condition	13	0.3%
DoS Flood	10	0.2%
CRLF Injection	8	0.2%
Eval Injection	8	0.2%
Numeric Error	7	0.1%

4855 total flaws tracked by CVE in 2008

The SANS/CWE Top 25 Programming Errors

CATEGORY: Insecure Interaction Between Components

1. CWE-20: Improper Input Validation
2. CWE-116: Improper Encoding or Escaping of Output
3. CWE-89: Failure to Preserve SQL Query Structure (aka 'SQL Injection')
4. CWE-79: Failure to Preserve Web Page Structure (aka 'Cross-site Scripting')
5. CWE-78: Failure to Preserve OS Command Structure (aka 'OS Command Injection')
6. CWE-319: Cleartext Transmission of Sensitive Information
7. CWE-352: Cross-Site Request Forgery (CSRF)
8. CWE-362: Race Condition
9. CWE-209: Error Message Information Leak

Visit cwe.mitre.org for details on each error

The SANS/CWE Top 25 Programming Errors

CATEGORY: Risky Resource Management

10. CWE-119: Failure to Constrain Operations within the Bounds of a Memory Buffer
11. CWE-642: External Control of Critical State Data
12. CWE-73: External Control of File Name or Path
13. CWE-426: Untrusted Search Path
14. CWE-94: Failure to Control Generation of Code (aka 'Code Injection')
15. CWE-494: Download of Code Without Integrity Check
16. CWE-404: Improper Resource Shutdown or Release
17. CWE-665: Improper Initialization
18. CWE-682: Incorrect Calculation

Visit cwe.mitre.org for details on each error

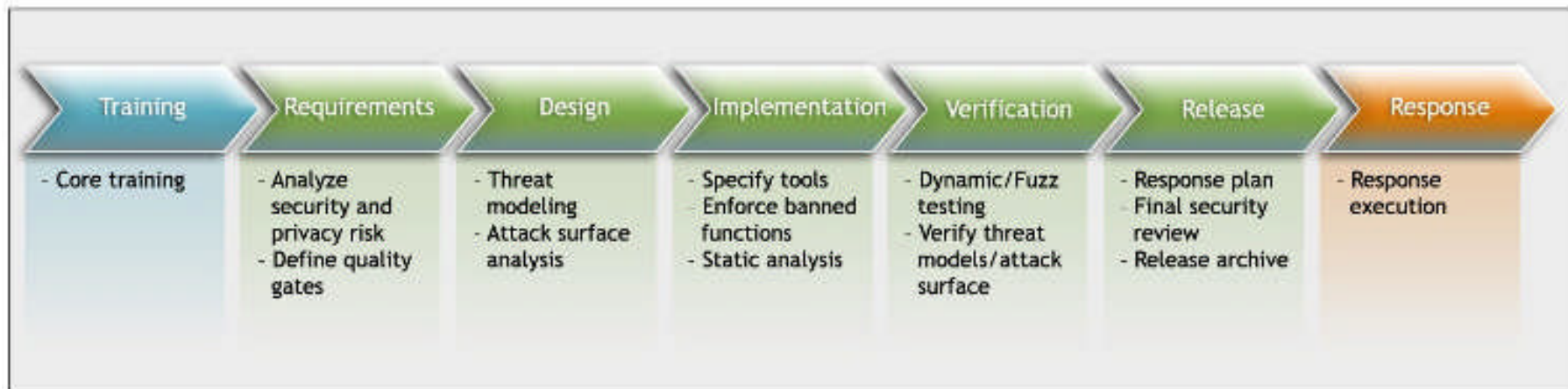
The SANS/CWE Top 25 Programming Errors

CATEGORY: Porous Defenses

19. CWE-285: Improper Access Control (Authorization)
20. CWE-327: Use of a Broken or Risky Cryptographic Algorithm
21. CWE-259: Hard-Coded Password
22. CWE-732: Insecure Permission Assignment for Critical Resource
23. CWE-330: Use of Insufficiently Random Values
24. CWE-250: Execution with Unnecessary Privileges
25. CWE-602: Client-Side Enforcement of Server-Side Security

Visit cwe.mitre.org for details on each error

Incorporating Security into the development lifecycle



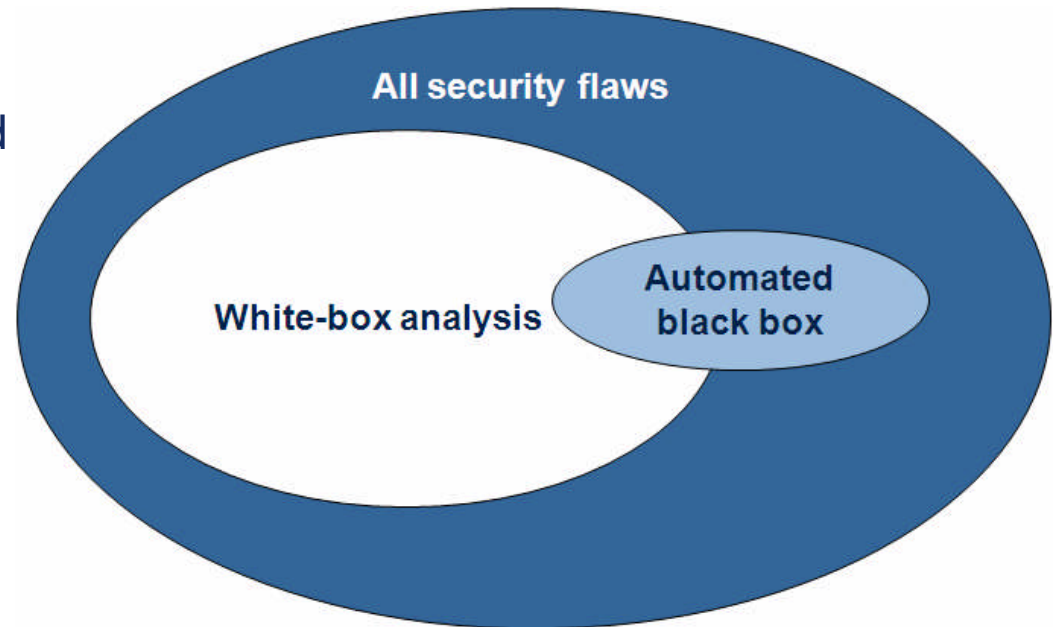
SANS Top 25 Mapped to Application Security Methods

CWE	Title	Education?	Manual Process?	Tools?	Threat Model?
20	Improper Input Validation	Y	Y	Y	Y
116	Improper Encoding or Escaping of Output	Y	Y	Y	
89	Failure to Preserve SQL Query Structure (aka SQL Injection)	Y	Y	Y	
79	Failure to Preserve Web Page Structure (aka Cross-Site Scripting)	Y	Y	Y	
78	Failure to Preserve OS Command Structure (aka OS Command Injection)	Y		Y	
319	Cleartext Transmission of Sensitive Information	Y			Y
352	Cross-site Request Forgery (aka CSRF)	Y		Y	
362	Race Condition	Y			
209	Error Message Information Leak	Y	Y	Y	
119	Failure to Constrain Memory Operations within the Bounds of a Memory Buffer	Y	Y	Y	
642	External Control of Critical State Data	Y			Y
73	External Control of File Name or Path	Y	Y	Y	
426	Untrusted Search Path	Y		Y	
94	Failure to Control Generation of Code (aka 'Code Injection')	Y	Y		
494	Download of Code Without Integrity Check				Y
404	Improper Resource Shutdown or Release	Y		Y	
665	Improper Initialization	Y		Y	
682	Incorrect Calculation	Y		Y	
285	Improper Access Control (Authorization)	Y	Y		Y
327	Use of a Broken or Risky Cryptographic Algorithm	Y	Y	Y	
259	Hard-Coded Password	Y	Y	Y	Y
732	Insecure Permission Assignment for Critical Resource	Y	Y		
330	Use of Insufficiently Random Values	Y	Y	Y	
250	Execution with Unnecessary Privileges	Y	Y		Y
602	Client-Side Enforcement of Server-Side Security	Y			Y








Source: 2009 Microsoft

Multiple Techniques Address the Universe of Security Flaws

- Universe of application security vulnerabilities is extensive
- Each technique has strengths and weaknesses
- There is no “silver bullet”
- A complete analysis includes:
 - Static Analysis (i.e. White Box)
 - Dynamic Analysis (i.e. Black Box)
 - Manual Penetration Testing

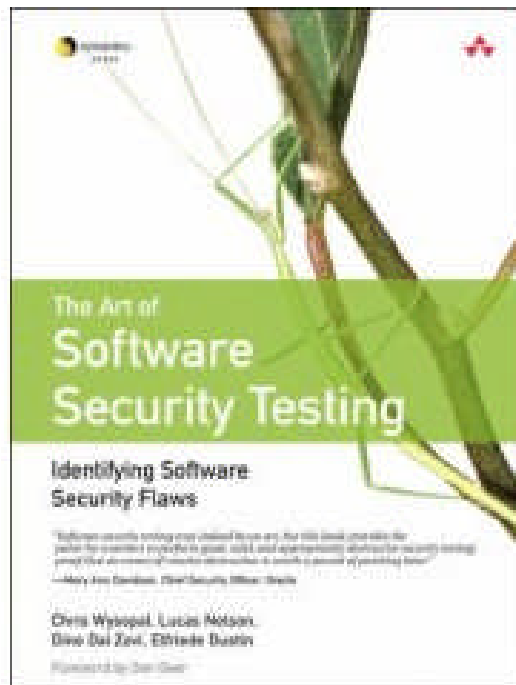


Automated Static vs. Dynamic Tradeoffs

	Static	Dynamic
Code Coverage		
Ease of Testing		
Noise		
Ratio of Actionable Vulnerabilities		
Cost		
Platform/Language Support		

Summary

- Application vulnerabilities is a growing area of risk
- Top 25 needs to be understood by development team and tested for before shipping or deploying code
- Combination of automated static, automated dynamic, and manual testing can detect these errors



**The Art of Software Security Testing:
Identifying Software Security Flaws**
by Chris Wysopal, Lucas Nelson, Dino Dai
Zovi, Elfriede Dustin

Q&A

cwysopal@veracode.com

VERACODE