Security Measurement and Assuring Reliability through metrics Technology (SMART)

Applying Reliability Metrics to Security Vulnerabilities

Wayne Zage, Dolores Zage, Blake Self
Presentation Outline

1. Background/overview of $S^2$ERC and design metrics
2. Vulnerability Analyses
Security and Software Engineering Research Center (S²ERC)

• An NSF Industry/University Cooperative Research Center established in 1986 and extended in 2010

• Participation by ten+ universities representing >50 researchers

• Collaborative, customized projects and technology transfer to affiliates are the hallmarks of the S²ERC
S²ERC Participating Universities

- Ball State
- DePaul
- Illinois at Chicago
- Iowa State
- IUPUI
- IPFW
- Penn State
- Purdue
- West Florida
- Limerick
S²ERC Industrial Affiliates

- Angie’s List
- Bingham McHale
- Blue Cross Blue Shield
- Boeing
- John Deere
- Intelligent Information Technologies
- Iowa Dept. of Transportation
- Lockheed Martin
- MacAulay Brown
- NASA
- Northrop Grumman
- Ontario Systems
- Raytheon
- NSWC – Crane
- Rockwell Collins
- TIAA
- Union Pacific
- US Army Research Lab
- US Dept. of Homeland Security
The S²ERC Design Metrics Research Timeline

- **1986**: Industry Development
  - Industry Validation Studies
  - Industry Validation - Northrop Grumman, Motorola, Bellcore, Raytheon, Army

- **1989**: Technology Development
  - Published Papers

- **1993**: Industry
  - Industry SE Symposium

- **1999**: SDL
  - Reliability Study

- **2003**: High Reliability Process Development

- **2004**: Ada, C, SDL, C++, Java, UML

- **2005**: Industry
  - Industry Mission Assurance Forum
  - Industry Workshops

- **2006**: Industry
  - Industry Workshops
  - NASA Symposium

- **2007**: Industry
  - Industry Workshops
  - NASA Workshops

- **2008**: Industry
  - Industry Mission Assurance Workshops
  - SERC Technical Report

- **2009**: Industry funded SERC Research
  - Industry funded SERC Research
  - NASA Symposium

- **2010**: Industry
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2011**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2012**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2013**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2014**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2015**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2016**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2017**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2018**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2019**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2020**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2021**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2022**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2023**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2024**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10

- **2025**: Industry funded SERC Research
  - Industry Workshops
  - NASA Workshops
  - ISSRE 05
  - SSTC 10
The 2007 Alexander Schwarzkopf Prize for Technological Innovation from the NSF I/UCRC Association recognized the achievements in developing software design metrics that identify fault-prone modules early in the software lifecycle, thereby allowing significant improvements in software quality and productivity.
Overview of the Design Metrics, $D_e$ and $D_i$

$D_e$ - an external view of design complexity
$D_i$ - an internal view of design complexity
Finding Outliers

S²ERC Design Metrics Research has been funded by ...

National Science Foundation
Motorola Corp.
Nortel Technologies
Telecordia Technologies
Northrop Grumman Corp.
Computer Sciences Corp.
GTE Data Services
Magnavox Electronic Systems Co.
Harris Corp.
Raytheon
US Army Research Lab
Ball State University
The design metrics have been computed on

university-based projects
CSC’s STANFINS project
systems from the US Army Research Lab
Harris’ ROCC project
Magnavox’s AFATDS project
PBX system from Telcordia Technologies
three Northrop Grumman projects
three Raytheon projects
telecommunications systems from Motorola
Results:
The design metrics have correctly identified at least 76% of the defect-prone modules 100% of the time.
The External Metric $D_e$

$D_e = e_1(\text{inflows} \times \text{outflows}) + e_2(\text{fan in} \times \text{fan out})$

$D_e = ((2+2+3) \times (1+2)) + (1 \times 2) = 23$
The External Metric $D_e$

$D_e = e_1 (\text{inflows} \times \text{outflows}) + e_2 (\text{fan in} \times \text{fan out})$

$$D_e = ((2+2+3) \times (1+2)) + (1 \times 2) = 23$$
The External Metric $D_e$

$D_e = e_1 (\text{inflows} \times \text{outflows}) + e_2 (\text{fan in} \times \text{fan out})$

$D_e = ( (2+2+3) \times (1+2) ) + (1 \times 2) = 23$
The External Metric $D_e$

$D_e = e_1(\text{inflows} \times \text{outflows}) + e_2 (\text{fan in} \times \text{fan out})$

$D_e = ( (2+2+3) \times (1+2) ) + (1\times2) = 23$
The Internal Metric $D_i$

$$D_i = w_1(CC) + w_2(DSM) + w_3(I/O)$$

where:

- **CC** (Central Calls) are procedure or function invocations
- **DSM** (Data Structure Manipulations) are references to complex data types
- **I/O** (Input/Output) are external device accesses
Extending Design Metrics Technology to a Software Security Engineering Process
Software Reality

Consists of complex, multiple technologies with multiple suppliers
Defects

Software Vulnerabilities

Unintentional Vulnerabilities

Intentional Vulnerabilities

EXPLOITABLE SOFTWARE

Exploitation potential of vulnerability independent of “intent”
Reliability and Security Parallels

Reliability context
(well-established) –
Design Metrics Reliability Research

Fault-prone component
Likely to contain faults

Failure-prone component
Likely to have failures in field

Security context
(to be established)
the SMART project funded by ARL

Vulnerability-prone component
Likely to contain vulnerabilities

Attack-prone component
Likely to be exploited in the field
SMART Project Objectives

- Investigate the overlap and interrelationships in the software constructs that affect the reliability and security of software
- Develop security metrics to identify, categorize, and analyze security weaknesses
Our Representative Systems/Technologies

- OpenSolaris
- Firefox
- OpenSSH
- httpd
- Drupal
Security: Expectations and Directions

• Some vulnerable components will be identified by design metrics.
• A huge “win” if 50% of the components containing documented vulnerabilities are identified by design metrics.
• New security-related primitives will be needed to increase that percentage.
Our Team’s General Process of Metric/Vulnerability Analysis

1. **Selected System’s Published CVS Reports**
   - **Scan** vulnerability updates

2. **Vulnerability Updates by module**
   - **Parses** results/reports analysis

3. **Selected System**
   - **Convert** XML representation

4. **XML representation**
   - **Collect** metrics on modules

5. **Metrics + Vulnerabilities**
   - **Match**

6. **Results/reports**
   - **Analysis**
# Common Weakness Enumeration and Systems

<table>
<thead>
<tr>
<th>CWE Category</th>
<th>OPEN SOLARIS</th>
<th>FIREFOX 2.0.0.1</th>
<th>FIREFOX 2.0.0.2</th>
<th>FIREFOX 2.0.0.5</th>
<th>Open-SSH</th>
<th>HTTPD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Quality</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>Data Handling</td>
<td>5</td>
<td>10</td>
<td>30</td>
<td>69</td>
<td>13*</td>
<td>19</td>
</tr>
<tr>
<td>Security Features</td>
<td>8</td>
<td>2</td>
<td>0</td>
<td>20</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Time and State</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Error Handling</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>API Abuse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>18</td>
<td>30</td>
<td>91</td>
<td>20</td>
<td>37</td>
</tr>
</tbody>
</table>
Secret life of the open source code
Ongoing Work

- Apache
- Open Solaris
- FireFox
- OpenSSH
- Drupal
Apache HTTP Server

- Designed by Robert McCool
- Developed by Apache Software foundation
- Initial release 1995
- Latest release 2.2.15 on 03/06/2010

- Since April 1996 Apache has been the most popular HTTP server on the WWW.
- As of March 2009 Apache served over 46% of all websites and over 66% of the million busiest.
Apache Vulnerabilities

- **Apache**
- For version 1.3.1
  - 144 files
  - 8 vulnerabilities identified
**D_e Metric Analysis on Apache 1.3.1**

87.5% or 7 out of the 8 vulnerable modules were identified in the top 10%

---

<table>
<thead>
<tr>
<th>Type</th>
<th>Number</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modules</td>
<td>1460</td>
<td>N/A</td>
</tr>
<tr>
<td># in the set for calculation</td>
<td>1460</td>
<td>N/A</td>
</tr>
<tr>
<td>Modules Highlighted</td>
<td>146</td>
<td>10%</td>
</tr>
</tbody>
</table>

---

**Weighting Scheme**

- $e_1 = 1.00$
- $e_2 = 1.00$

---

**Analysis Table**

- Cases: 1460
- Max: 35248
- Min: 0
- Mean: 246.42
- Stddev: 1346.29
- Percentage: 10
- Sum: 359771
Graphic of Indicated Stress Modules
OpenSolaris

- based on Sun Microsystems' Solaris
- Latest release 2009.06 on 06/01/2009
- OpenSolaris is derived from the Unix System V Release 4 codebase, with significant modifications made by Sun since it bought the rights to that code in 1994.

OpenSolaris Vulnerabilities
OpenSolaris Vulnerabilities and Source

- 22,600 files in the downloadable tar file
- 23 module updates from vulnerabilities totaling 37 changes
- 5 is the largest # of changes on one module
- We want to identify the 23 modules out of the 22,600 x modules in files, approximately 1 in 10,000
Divide and Conquer

- 18 module vulnerability updates located in the /usr/src/uts/common stem (3983 files)
- 5,946,281 xml tags in the 3983 files
- 5 module vulnerability updates located in the /usr/src/cmd/sgs/rtld/common stem (29 files)
- 90,417 xml tags in the 29 files
D\(_e\) Metric Analysis on 
/usr/src/cmd/sgs/rtld/common
60% or 3 out of the 5 vulnerable modules or
69% or 9 out of 13 changes were identified
FireFox

• Mozilla Firefox is a web browser
• Designed by and developed by Mozilla Corporation
• Initial release November 9, 2004
• Latest release 3.6.6 on 06/26/2010

• Firefox had 22.05% of the recorded usage share of web browsers as of March 2009, making it the second most popular browser in terms of current use worldwide, after Internet Explorer.

FireFox Vulnerabilities
General Analysis

• Multiple versions of three software systems studied
• Source Code is THE primary source of vulnerability reports
• At least 51% and at most 86% of reported vulnerabilities are in Data Handling
• Data Structure Manipulations (DSM) is the best predictor of vulnerabilities
OpenSSH

• Developed by OpenBSD Project
• OpenSSH first appeared in OpenBSD 2.6 and the first portable release was made in October 1999
• Latest release 5.5/5.5p1 on 04/16/2010

OpenSSH Vulnerabilities
OpenSSH 3.8p1

- Files: 243
- Modules: 2,437
- Definitions: 2,992
- preprocessor directives: 5,147
- user defined include files: 1,101
- conditional expressions: 61,815
- xml tags: 703,850
- **31** vulnerable modules identified

- Ranked modules by De
- 18 of the 31 vulnerable modules in the top 10% (58%)
- 23 of the 31 vulnerable modules in top 20% (74%)
- The remaining 8 had additional suspicious patterns:

  Modules were named “x” and had a “mate” x1
  Duplicate names
Drupal

- Initial release January 2001 (2001-01)
- Latest release 6.17 June 2010
- Written in PHP
- Operating system cross-platform
  - Open Source Content Management System (CMS)
  - Over 350,000 subscribed members today
  - Consists of PHP, INC, JavaScript, Perl, XML files
  - DrupalSites.net is a directory that list thousands of websites powered by Drupal
  - Winner of Best Overall 2008 Open Source CMS Award for Second Year in a Row
  - Listed as one of the Open Source PHP applications that changed the world
Drupal Vulnerabilities and Source

• PHP Process of Metric/Vulnerability Analysis
  • Constructed the **Drupal Vulnerabilities Miner** (DVM)
  • the Drupal CVS web site ([http://www.drupal.org/security](http://www.drupal.org/security)).
  • DVM isolated 277 RCS file patches identified from 140 vulnerability updates.
• Approximately 105 Drupal PHP files
Generic Source Analysis

Selected System → convert → XML representation → collect → Metrics On Modules

PHP Source Analysis

Designing a PHP2XML tool

Mapping PHP primitive tags to primitive design metrics
Benefits

• A knowledge of *where* vulnerabilities are most likely to reside can help prioritize security efforts.

• Analyzing multiple technologies and mapping the vulnerabilities to the CWE to ensure coverage

• Merge multiple technologies through mapping the individual xml representations, isolates commonality and individuality
Next Research Steps

• Continue the analysis of the open source systems
• Investigate other primitives to identify, categorize security weakness
• Apply technologies to systems other than open source (YOURS?)
• Combine into the network model