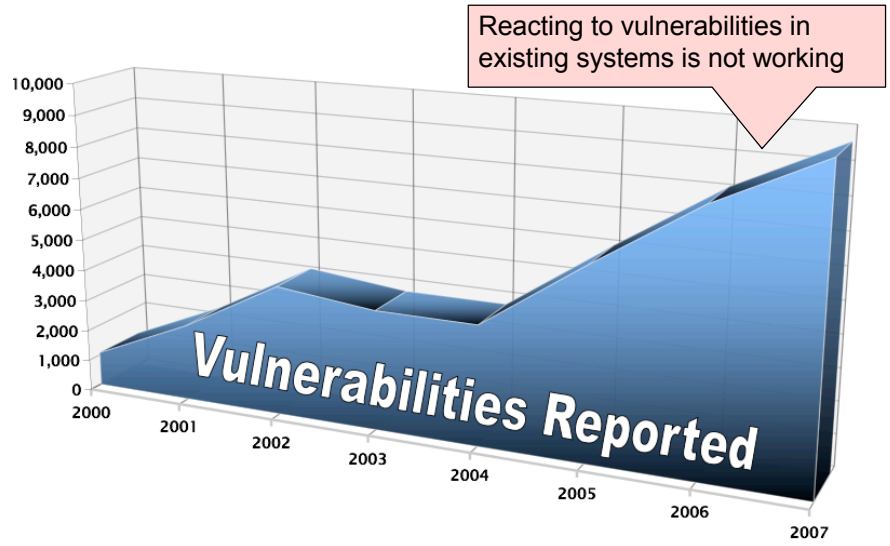
Secure Coding Initiative

Jason A. Rafail
Monday, May 14th, 2007

© 2007 Carnegie Mellon University

Increasing Vulnerabilities



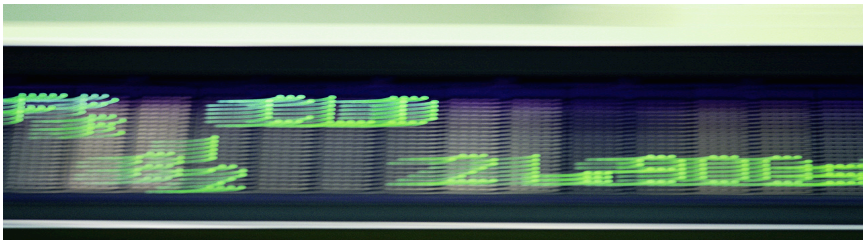
Vulnerabilities Cost Vendors

A study based on real vulnerability announcements in 1999-2004 revealed: an average **drop** of the concerned vendor's **stock price** of **0.6%** after each vulnerability announcement

– *Tehang / Wattal, Carnegie Mellon Univerisity, 2004*

"Impact of Software Vulnerability Announcements on the Market Value of Software Vendors – an Empirical Investigation"

... not to mention the damage to the vendor's reputation



Most Vulnerabilities caused by Programming Errors

64% of the vulnerabilities in NVD in 2004 are due to programming errors

- 51% of those due to classic errors like buffer overflows, cross-site-scripting, injection flaws
- *Heffley/Meunier (2004): Can Source Code Auditing Software Identify Common Vulnerabilities and Be Used to Evaluate Software Security?*

Cross-site scripting, SQL injection at top of the statistics (CVE, Bugtraq) in 2006

"We wouldn't need so much network security if we didn't have such bad software security"

--Bruce Schneier

Unexpected Integer Values

An unexpected value is not one you would expect to get using a pencil and paper

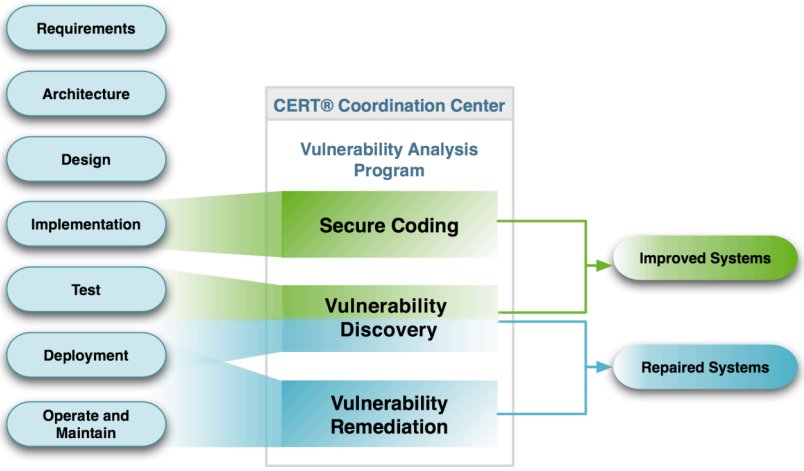
Unexpected values are a common source of software vulnerabilities (even when this behavior is correct).

Fun With Integers

```
char x, y;
x = -128;
y = -x;

if (x == y) puts("1");
if ((x - y) == 0) puts("2");
if ((x + y) == 2 * x) puts("3");
if (x != -y) puts("4");
```

CERT Vulnerability Analysis



CERT Secure Coding Initiative

Work with software developers and software development organizations to eliminate vulnerabilities resulting from coding errors before they are deployed.

- Reduce the number of vulnerabilities to a level where they can be handled by computer security incident response teams (CSIRTs)
- Decrease remediation costs by eliminating vulnerabilities *before* software is deployed

Overall Thrusts

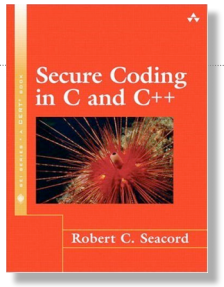
Advance the state of the practice in secure coding

Identify common programming errors that lead to software vulnerabilities

Establish standard secure coding practices

Educate software developers

Current Capabilities



Secure Coding in C and C++

- Addison-Wesley book
- Training

Secure coding web pages
www.cert.org/secure-coding/

Secure string and integer library development

Involvement in international standards activities:

- ISO/IEC JTC1/SC22/WG14 C programming language international standardization working group
- ISO/IEC JTC1/SC22 OWG Vulnerabilities

Current and Planned Efforts

CERT Secure Coding Standards

- C and C++ Programming Language
- Community development process

Training courses

- Direct offerings
- Partnered with industry

Software Validation and Verification

- Partner with software tool vendors to validate conformance to secure coding standards
- Partner with software development organizations to evaluate the application of secure coding standards

CERT Secure Coding Standards

Identify coding practices that can be used to improve the security of software systems under development

Specific objectives include

- avoiding undefined behaviour
- avoiding implementation defined behaviour
- improving clarity for review and maintenance
- providing a consistent style across a program or set of programs
- avoiding common programmer errors
- incorporating good practice, particularly with regard to ‘future proofing’

Scope

The secure coding standards proposed by CERT are based on documented standard language versions as defined by official or *de facto* standards organizations.

Secure coding standards are under development for:

- C programming language (ISO/IEC 9899:1999)
- C++ programming language (ISO/IEC 14882-2003)

Applicable technical corrigenda and documented language extensions such as the ISO/IEC TR 24731 extensions to the C library are also included.

Secure Coding Web Site (Wiki)

Secure Coding

CERT Secure Coding Standards

Added by Confluence Administrator, last edited by Robert Seacord on Feb 28, 2007 (view change)
Labels: (None)

Welcome to the Secure Coding Web Site

This web site exists to support the development of secure coding standards for commonly used programming languages such as C and C++. These standards are being developed through a broad-based community effort including the CERT Secure Coding Initiative and members of the software development and software security communities. For a further explanation of this effort please read the Rationale.

As this is a development web site, many of the pages on this web site are incomplete or contain errors. If you are interested in furthering this effort you may comment on existing items or send recommendations to secure-coding at cert dot org. You may also apply for an account to directly edit content on the site. Before using this site, please familiarize yourself with the Terms and Conditions.

The Top 10 Secure Coding Practices provides some language independent recommendations.

Secure Coding Standards

CERT C Programming Language Secure Coding Standard

CERT C++ Programming Language Secure Coding Standard

We would like to acknowledge the contributions of the following folks, and we look forward to seeing your name there as well.

Rules and Recommendations

Coding practices are classified as either rules or recommendations

- Rules need to be followed to claim compliance.
- Recommendations are guidelines or suggestions.

Rules

Coding practices are defined as rules when

- Violation of the coding practice will result in a security flaw that may result in an exploitable vulnerability.
- There is an enumerable set of exceptional conditions (or no such conditions) where violating the coding practice is necessary to ensure the correct behavior for the program.
- Conformance to the coding practice can be verified.

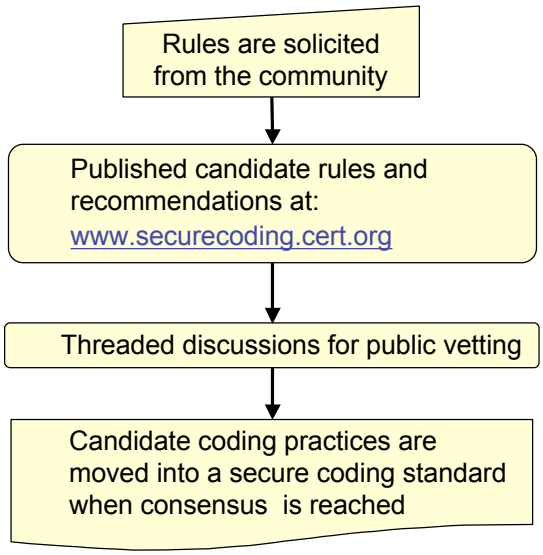
Recommendations

Coding practices are defined as recommendations when

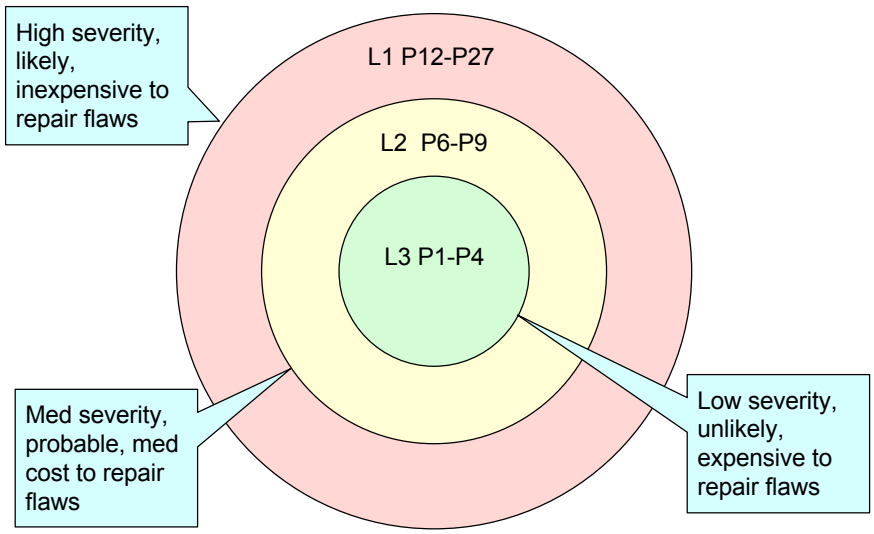
- Application of the coding practice is likely to improve system security.
- One or more of the requirements necessary for a coding practice to be considered as a rule cannot be met.

Community Development Process

Secure coding standards development is a community effort



Priorities and Levels



Rules

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
FIO30-C	**3** (high)	**3** (probable)	**3** (low)	**P27**	**L1**
FIO32-C	**3** (high)	**2** (probable)	**1** (medium)	**P6**	**L2**
FIO33-C	**1** (low)	**1** (low)	**3** (medium)	**P3**	**L3**
FIO34-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO35-C	**1** (low)	**1** (unlikely)	**2** (medium)	**P2**	**L3**
FIO36-C	**1** (low)	**1** (unlikely)	**3** (low)	**P3**	**L3**
FIO37-C	**3** (high)	**1** (unlikely)	**2** (medium)	**P6**	**L3**
FIO38-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO39-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO40-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO41-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO42-C	**2** (medium)	**2** (probable)	**2** (medium)	**P8**	**L2**
FIO43-C	**3** (high)	**3** (likely)	**2** (low)	**P18**	**L1**

Applications

Establish secure coding practices within an organization

- may be extended with organization-specific rules
- cannot replace or remove existing rules

Train software professionals

Certify programmers in secure coding

Establish base-line requirements for software analysis tools

Certify software systems

For More Information

Visit CERT® web sites:

http://www.cert.org/secure-coding/
https://www.securecoding.cert.org/

Contact Presenter

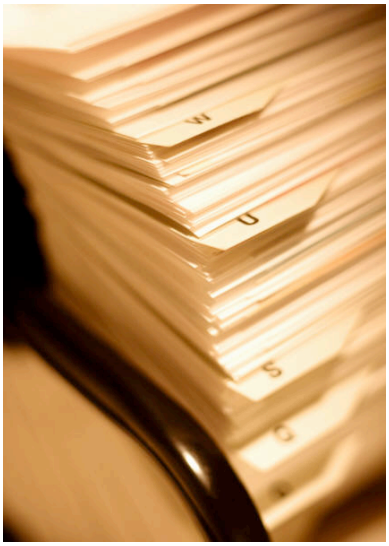
Jason A. Rafail jrafail@cert.org

Contact CERT Coordination Center:

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh PA 15213-3890
USA

Hotline: **+1 412 268 7090**

CERT/CC personnel answer 24x7, 365 days per year

Fax: **+1 412 268 6989**

Mailto: **cert@cert.org**