Software Assurance: Crippling Coming Cyberassaults

Dr. Paul E. Black

National Institute of Standards and Technology

http://samate.nist.gov/

paul.black@nist.gov

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
APR 2010		**00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Software Assurance: Crippling Coming Cyberassaults	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **National Institute of Standards and Technology,100 Bureau Dr # 1000 ,Gaithersburg,MD,20899**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT.

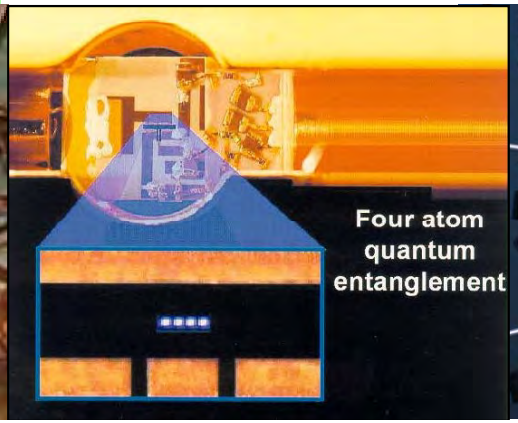
14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**29**	

What is NIST?

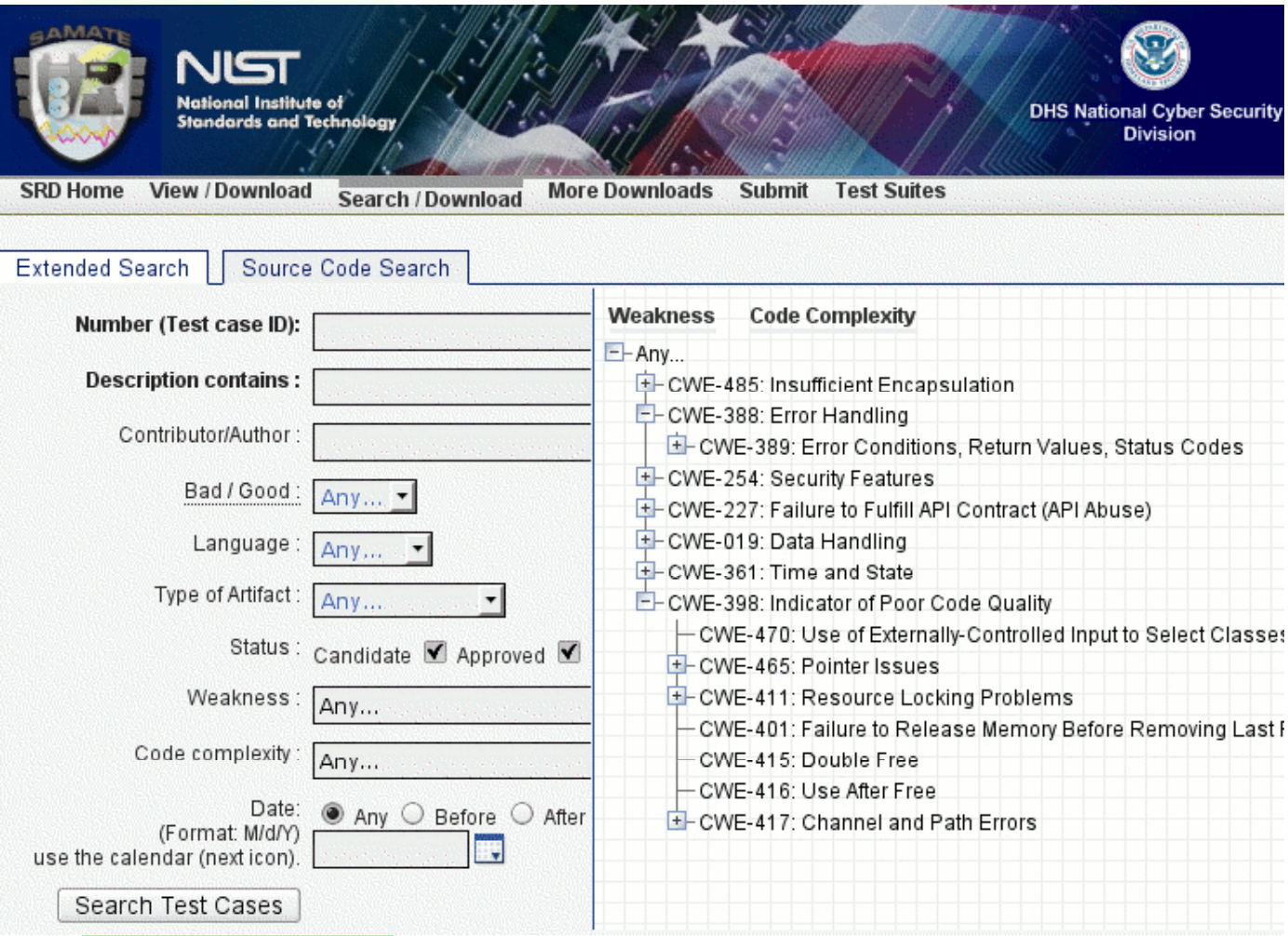
- **U.S. National Institute of Standards and Technology**
- **A non-regulatory agency in Dept. of Commerce**
- **3,000 employees + adjuncts**
- **Gaithersburg, Maryland and Boulder, Colorado**
- **Primarily research, not funding**
- **Over 100 years in standards and measurements: from dental ceramics to microspheres, from quantum computers to fire codes, from body armor to DNA forensics, from biometrics to text retrieval.**



The NIST SAMATE Project

- **Software Assurance Metrics And Tool Evaluation (SAMATE) project is sponsored in part by DHS**
- **Current areas of concentration**
  - **Web application scanners**
  - **Source code security analyzers**
  - **Static Analyzer Tool Exposition (SATE)**
  - **Software Reference Dataset**
  - ***Software labels***
  - ***Malware research protocols***
- **Web site http://samate.nist.gov/**

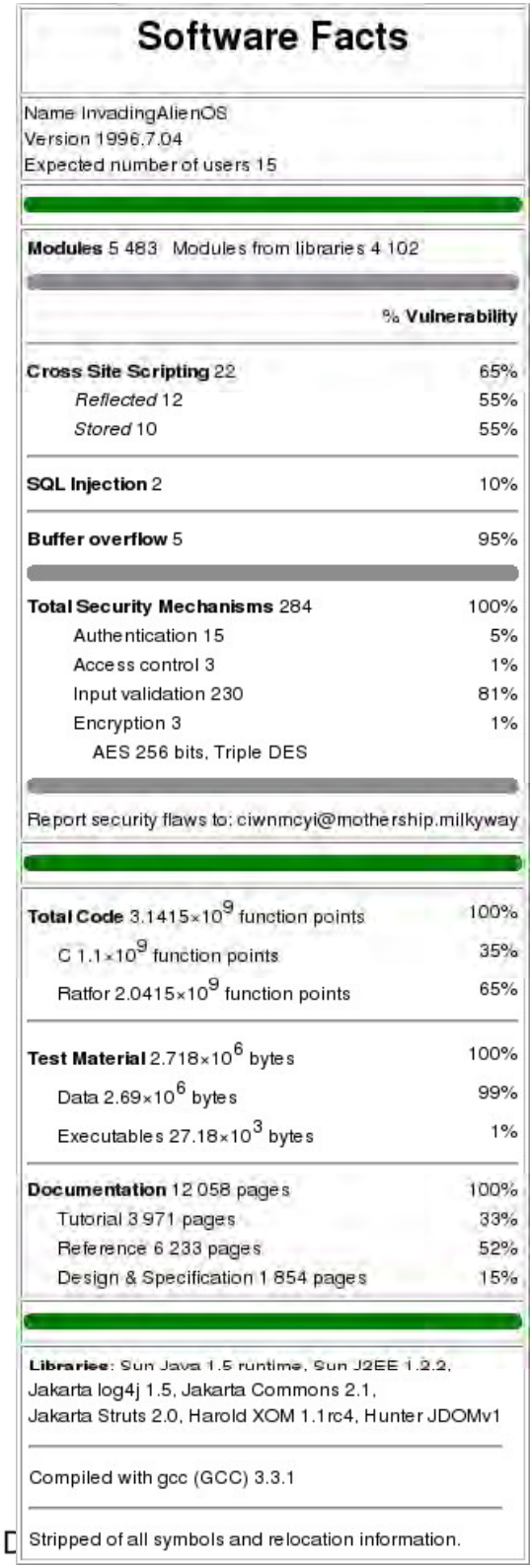
Software Reference Dataset



- **Public repository for software test cases**
- **Almost 1800 cases in C, C++, Java, and Python**
- **Search and compose custom Test Suites**
- **Contributions from Fortify, Defence R&D Canada, Klocwork, MIT Lincoln Laboratory, Praxis, Secure Software, etc.**

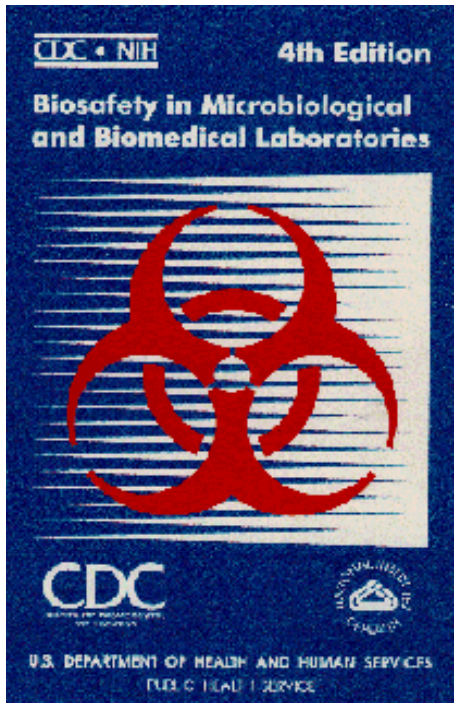
Software Facts Label

- **Software Facts should:**
  - **Voluntary**
  - **Absolutely simple to produce**
  - **Have a standard format for other claims**
- **What could be easily supplied?**
  - **Source available? Yes/No/Escrowed**
  - **Default installation is secure?**
  - **Accessed: network, disk, ...**
  - **What configuration files? (registry, ...)**
  - **Certificates (eg, "No Severe weaknesses found by CodeChecker ver. 3.2")**
- **Cautions**
  - **A label can give false confidence.**
  - **A label shut out better software.**
  - **Labeling diverts effort from real improvements.**



Researching Risky Software

- **Many people research malware, but there are no widely accepted protocols.**
- **Biological research has defined levels with associated practices, safety equipment, and facilities.**
- **Some approaches are**
  - **Weakened programs (auxotrophs)**
  - **Programs that ALERT**
  - **Outgoing firewalls**
  - **Isolated networks**



- **Assurance that software is less vulnerable to coming cyberassaults**
- **Static and dynamic analysis**
- **Static Analysis Tool Exposition - 2009 outcomes and 2010 progress**

Assurance from three sources

$$A = f(p, s, e)$$

where *A* is functional assurance, *p* is process quality, *s* is assessed quality of software, and *e* is execution resilience.

p is process quality

- **High assurance software must be developed with care, for instance:**
  - **Validated requirements**
  - **Good system architecture**
  - **Security designed- and built in**
  - **Trained programmers**

s is assessed quality of software

- **Two general kinds of software assessment:**
  - **Static analysis**
    - **e.g. code reviews and scanner tools**
    - **examines code**
  - **Testing (dynamic analysis)**
    - **e.g. penetration testing, fuzzing, and red teams**
    - **runs code**

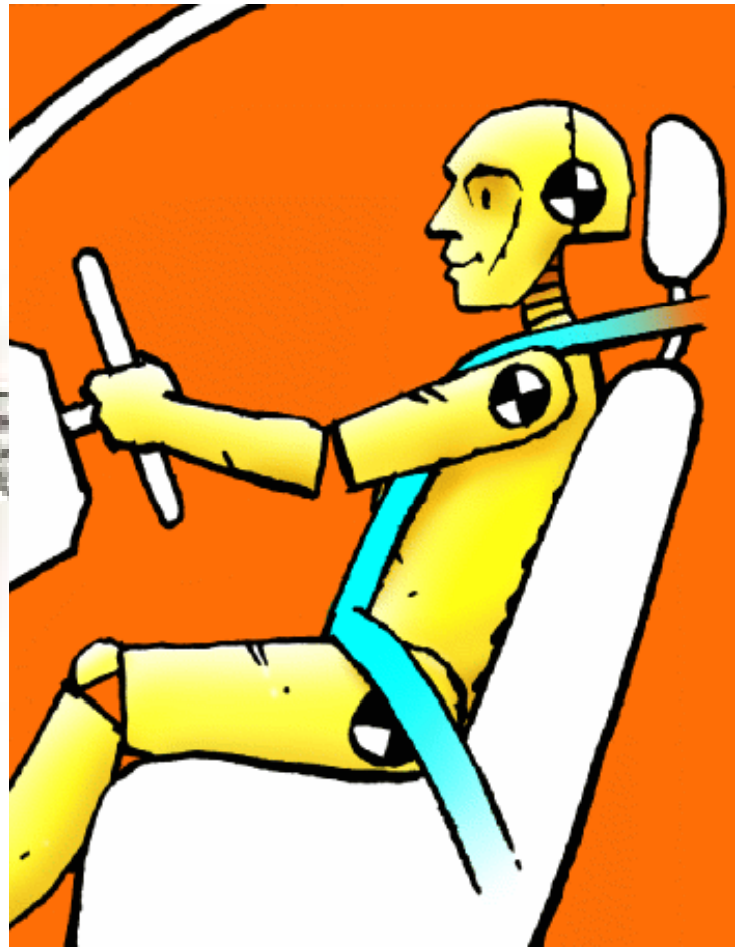
e is execution resilience

- **The execution platform can add assurance that the system will function as intended.**
- **Some techniques are:**
  - **Randomize memory allocation**
  - **Execute in a “sandbox” or virtual machine**
  - **Monitor execution and react to intrusions**
  - **Replicate processes and vote on output**

Software analysis is vital

- **Benefits are:**
  - **Provide feedback to development process**
  - **Build product assurance when process is less visible**
    - **contractors**
    - **open source**
    - **legacy software**
  - **Confirm minimum quality for execution**

Analysis is like a seatbelt …

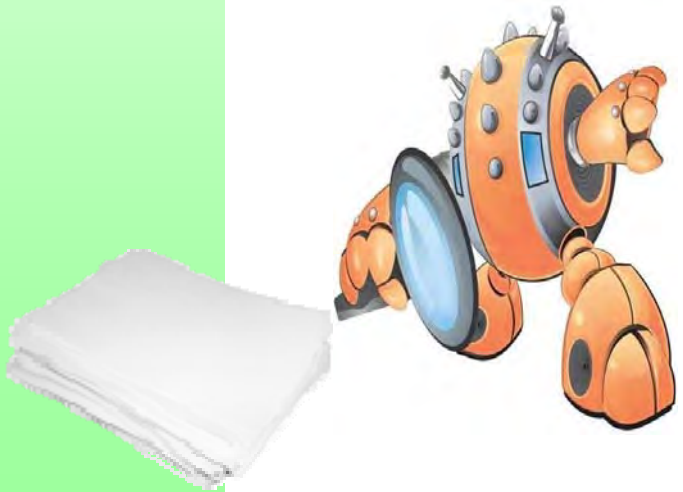


- Assurance that software is less vulnerable to coming cyberassaults
- **Static and dynamic analysis**
- Static Analysis Tool Exposition - 2009 outcomes and 2010 progress

Comparing Static Analysis with Dynamic Analysis

Static Analysis

- Code review
- Binary, byte, or source code scanners
- Model checkers & property proofs
- Assurance case



Dynamic Analysis

- Execute code
- Simulate design
- Fuzzing, coverage, MC/DC, use cases
- Penetration testing
- Field tests

Strengths of Static Analysis

- **Applies to many artifacts, not just code**
- **Independent of platform**
- **In theory, examines *all possible* executions, paths, states, etc.**
- **Can focus on a single specific property**

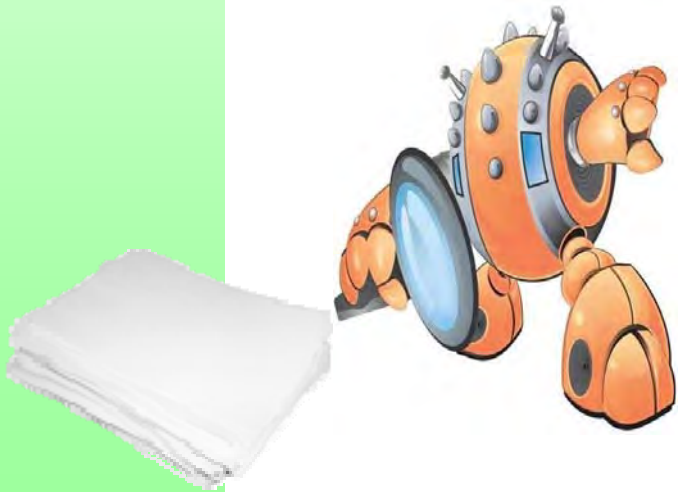
Strengths of Dynamic Analysis

- **No need for code**
- **Conceptually easier - “if you can run the system, you can run the test”.**
- **No (less) need to build or validate models or make assumptions.**
- **Checks installation and operation, along with end-to-end or whole-system.**

Static and Dynamic Analysis Complement Each Other

Static Analysis

- Handles unfinished code
- Can find backdoors, eg, full access for user name "JoshuaCaleb"
- Potentially complete



Dynamic Analysis

- Code not needed, eg, embedded systems
- Has few(er) assumptions
- Covers end-to-end or system tests
- Assess as-installed

- Assurance that software is less vulnerable to coming cyberassaults
- Static and dynamic analysis
- **Static Analysis Tool Exposition - 2009 outcomes and 2010 progress**

Static Analysis Tool Exposition (SATE) Overview

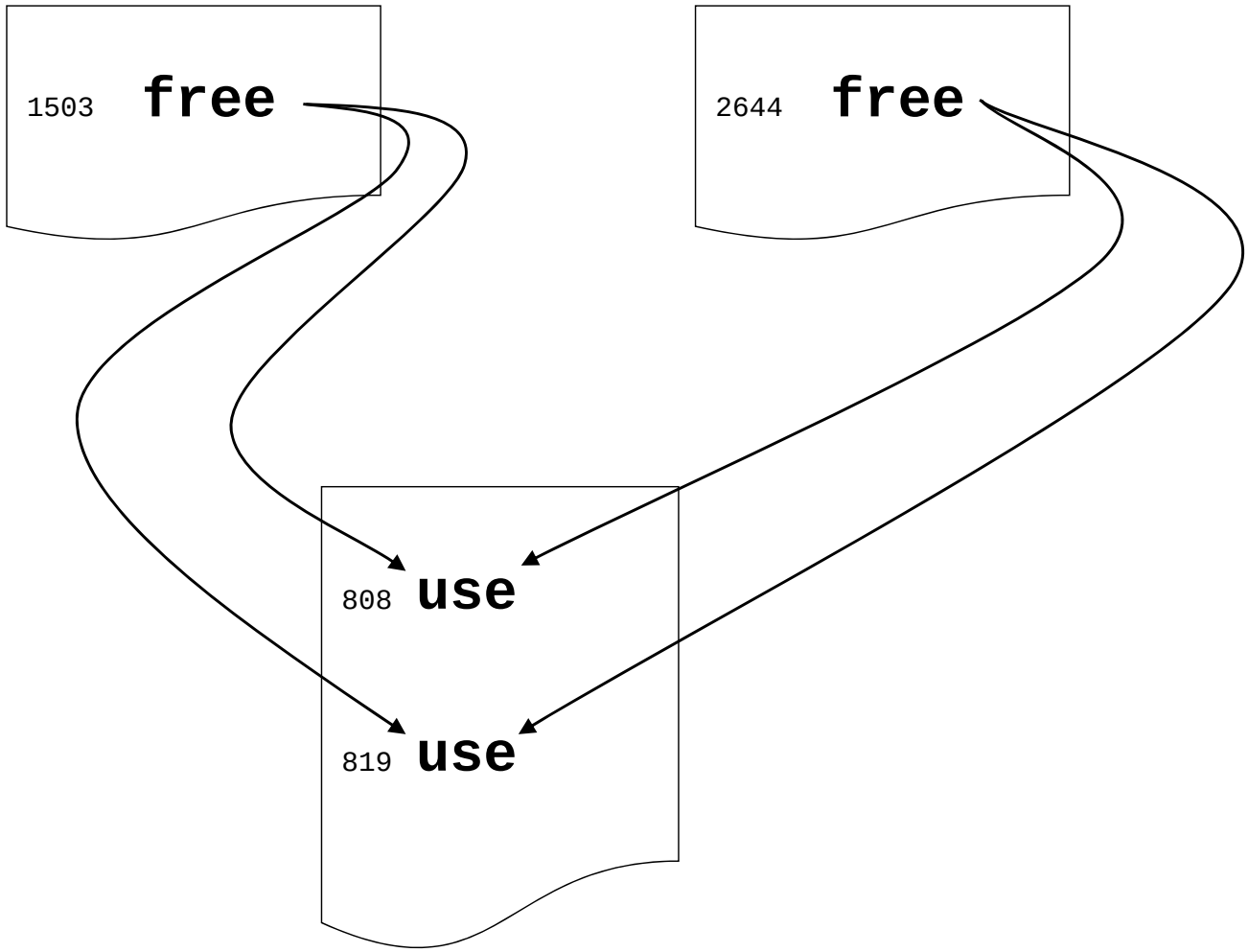
- **Goal: advance research in, and improvement of, static analysis tools for security-relevant defects and speed tool adoption by demonstrating use on real software.**
- **Checkpoints**
  - **Participants run tools on Java and C programs we choose**
  - **NIST-led researchers analyze reports**
  - **Everyone shares results and observations at a workshop**
  - **Later release final report and all data**
- **http://samate.nist.gov/SATE.html**
- **Co-funded by NIST and DHS/NCSD**

SATE Participants

- **2008:**
  - **Aspect Security ASC**
  - **Checkmarx CxSuite**
  - **Flawfinder**
  - **Fortify SCA**
  - **Grammatech CodeSonar**
  - **HP DevInspect**
  - **SofCheck Inspector for Java**
  - **UMD FindBugs**
  - **Veracode SecurityReview**
- **2009:**
  - **Armorize CodeSecure**
  - **Checkmarx CxSuite**
  - **Coverity Prevent**
  - **Grammatech CodeSonar**
  - **Klocwork Insight**
  - **LDRA Testbed**
  - **SofCheck Inspector for Java**
  - **Veracode SecurityReview**

“Number of bugs” is undefined

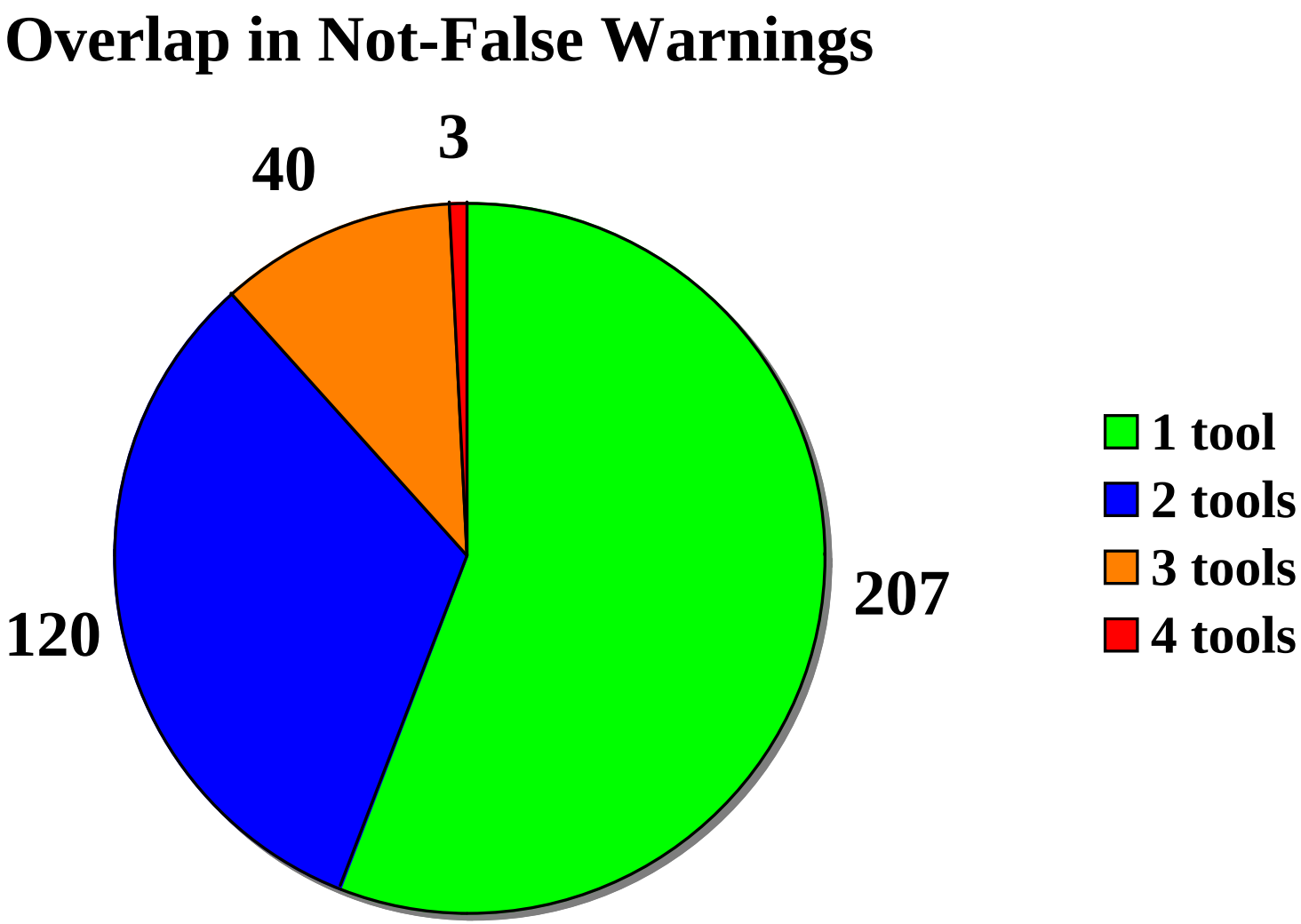
Tangled Flow: 2 sources, 2 sinks, 4 paths



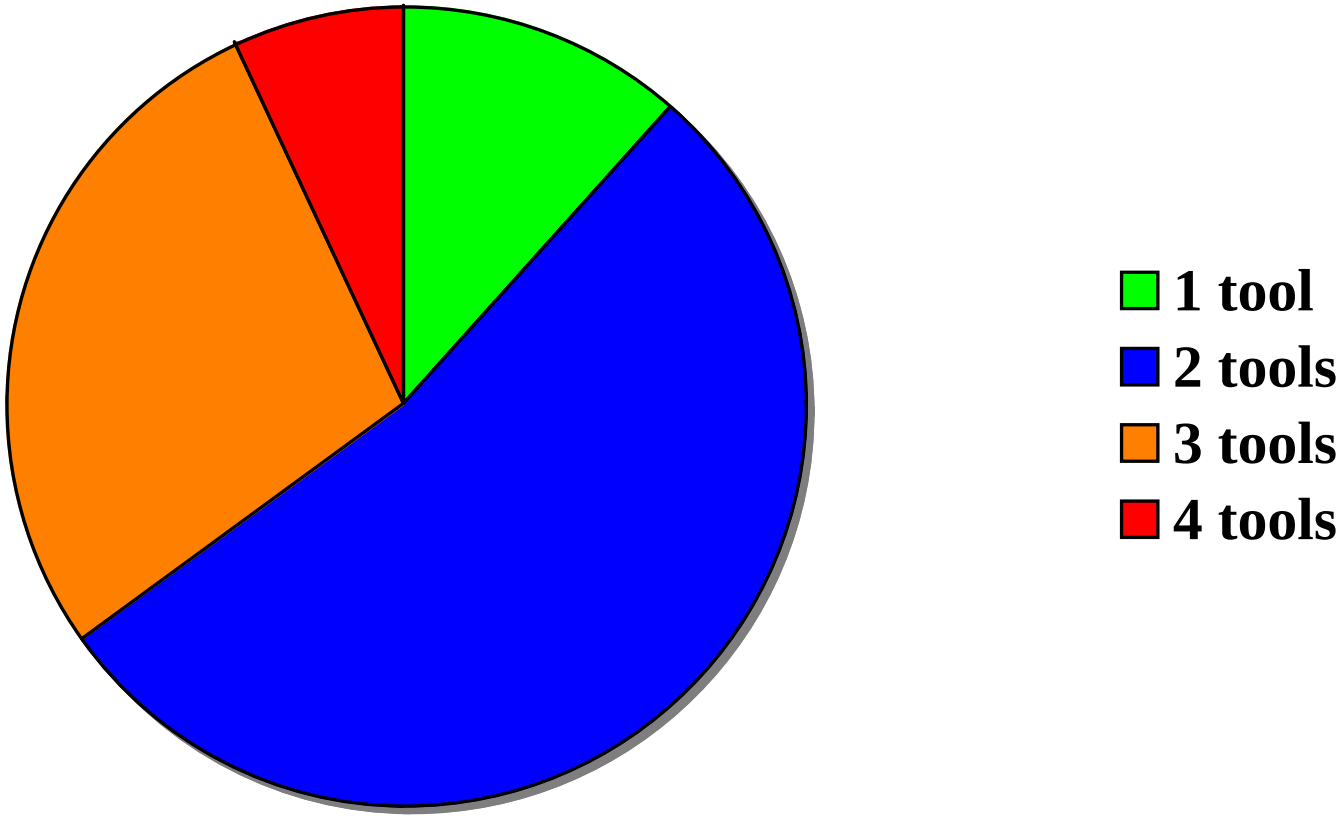
Summary of 2009 tool reports

- **Reports from 18 tool runs**
- **About 20,000 total warnings**
  - **but tools prioritize by severity, likelihood**
- **Reviewed 521 warnings - 370 were not false**
- **Number of warnings varies a lot by tool and case**
- **83 CWE ids/221 weakness names**

Tools don’t report same warnings

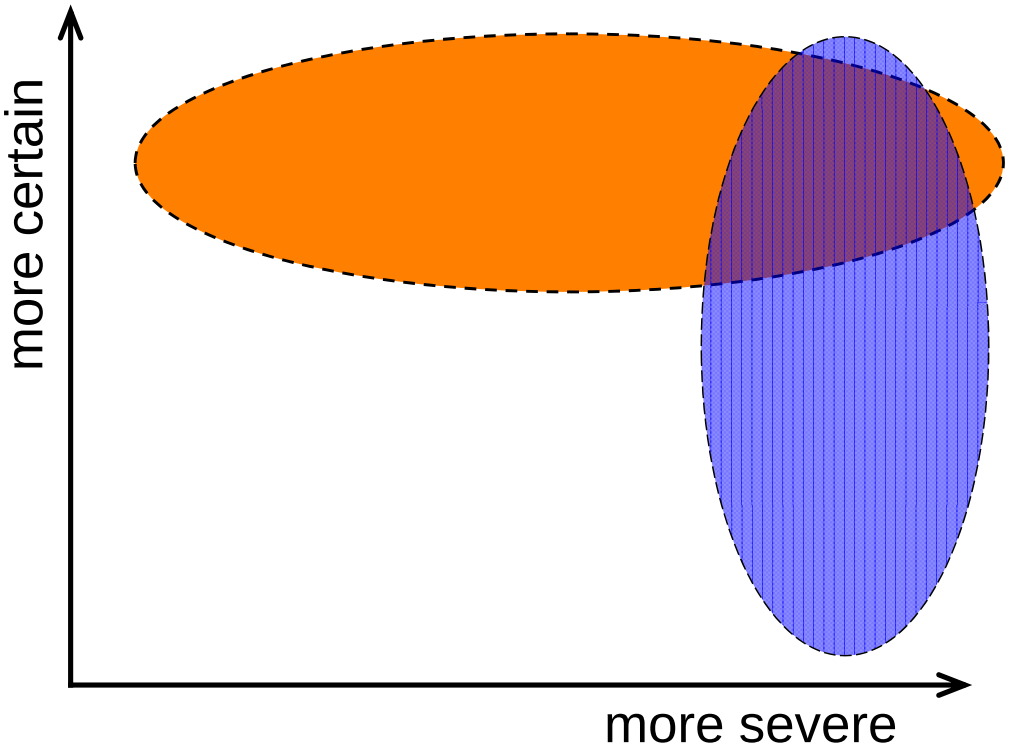


Some types have more overlap

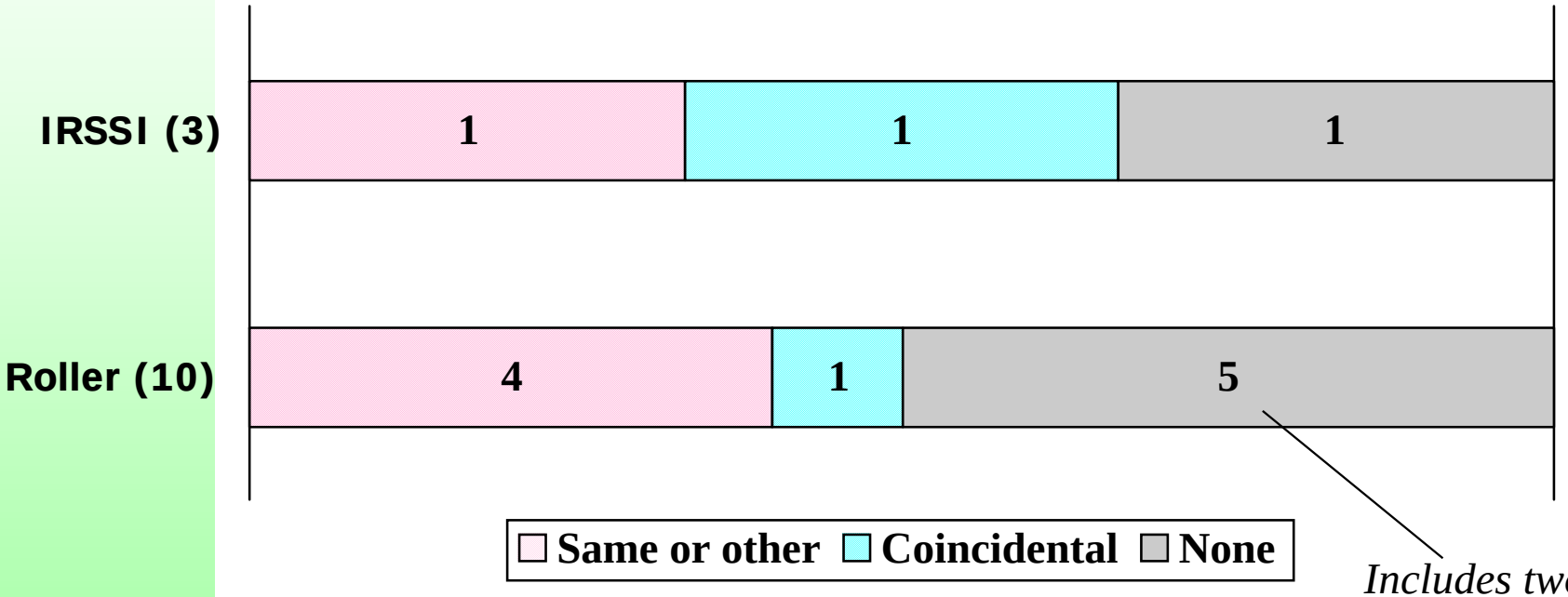


Why don’t tools find same things?

- **Tools look for different weakness classes**
- **Tools are optimized differently**



Tools find things that people find



SATE 2010 tentative timeline

- ✓ **Hold organizing workshop (12 Mar 2010)**
- ✓ **Recruit planning committee.**
- **Revise protocol.**
- **Choose test sets. Provide them to participants (17 May)**
- **Participants run their tools. Return reports (25 June)**
- **Analyze tool reports (27 Aug)**
- **Share results at workshop (October)**
- **Publish data (after Jan 2011)**

Acronyms

- **CWE - Common Weakness Enumeration**
  http://cwe.mitre.com/
- **DHS/NCSD - Department of Homeland Security/National Cyber Security Division**
- **MC/DC - Modified Condition/Decision Coverage**
- **SAMATE - Software Assurance Metrics And Tool Evaluation (project at NIST)**
- **SATE - Static Analysis Tool Exposition (annual event)**
- **NIST - National Institute of Standards and Technology**