SATE IV Background

Vadim Okun, NIST
vadim.okun@nist.gov
March 29, 2012

The SAMATE Project
http://samate.nist.gov/
Cautions on Using SATE Data

• Our analysis procedure has limitations
• In practice, users write special rules, suppress false positives, and write code in certain ways to minimize tool warnings
• There are many other factors that we did not consider: user interface, integration, etc.
• So do NOT use our analysis to rate/choose tools
Analyzing Source Code Analyzers


Program

Tool C
Tool B
Tool A
Buf
Leak
Race
...

National Institute of Standards and Technology
Warning Selection Methods

1. Random subset
2. Related to CVEs
3. Related to human findings
4. Synthetic test cases
SATE IV timeline

• Provide test sets to teams (31 July 2011)
• Teams run their tools, return reports (31 Oct)
• Analyze tool reports, with feedback from teams (12 March 2012)
• Experience sharing at workshop (here & now)
• Teams can submit a research paper (May)
• Publish data (Sep - Dec 2012)
Participating teams

- Buguroo BugScout
- Concordia University Marfcat
- Cppcheck
- Grammatech CodeSonar
- LDRA Testbed
- Monoidics INFER
- Parasoft C++test and Jtest
- Red Lizard Software Goanna
Test cases

• CVE-selected vulnerable/fixed pairs:
  – Dovecot: secure IMAP and POP3 server – C
  – Wireshark: network protocol analyzer – C
  – Tomcat: servlet container – Java
  – Jetty: servlet container – Java
    • All are open source programs
    • 96k LoC (Jetty) to 1.6M LoC (Wireshark)

• 59k synthetic C/C++ and Java test cases
Tool reports

• Teams converted reports to SATE format
  – SAFES format - optional
  – Some original reports
• Described environment in which they ran tool
• Some teams tuned their tools
• Some teams provided analysis of their tool warnings
Analysis procedure for CVE-selected test cases

Selection Methods:

- Tool warnings ~52K
- Selected randomly
- Related to CVEs
- To human findings
- Analyze for correctness and associate
- Selected warnings
- Analyze the data
Warning Subset Selection

*For vulnerable versions only*

- We assigned severity if a tool did not
- Avoid warnings with severity 5 (lowest)
- Statistically select from each warning class
- Select more warnings from higher severities
- Select 30 warnings from each of 15 tool reports
  - 1 report had only 6 warnings
  - Did not analyze Marfcat warnings
- Total is 426
Correctness categories

- True security weakness
- True quality weakness
- True but insignificant weakness
- Weakness status unknown
- Not a weakness
CVEs

• Identify the CVEs
  – Locations in code

• Find related warnings from tools

• Can tools discriminate between versions
  – Or report for a fixed version also?

• Goal: focus our analysis on real-life exploitable vulnerabilities
Human findings

For IPMI protocol of Wireshark only

• Security experts analyze test case
  – Mike Cooper and David Lindsay from Cigital
• Look for important weaknesses
  – Root cause, with an example trace
• Look for related warnings from tools
Analysis procedure for synthetic test cases

- Precisely characterized weaknesses
- Mechanical matching is not perfect

Tool warnings ~185K

Mechanically match warnings by name/CWE

In “bad” code? → TP
In “good” code? → FP

If no match: ignore
SATE over time

• 2008: First try: analyze warnings
• 2009: Subset selection, more analysis categories, human findings
• 2010: CVE-selected test cases, improved analysis guidelines
• IV: Added synthetic test cases
Differences from SATE 2010

- Synthetic test cases
- Same test cases for CVE-selected and sample analysis
- Describe CVEs better
- Test cases pre-compiled in a Virtual Machine
- More time to run tools, analyze outputs

- Still, much can be improved…
Thanks to teams!