Software Fault Interactions

Rick Kuhn
National Institute of Standards & Technology
Kuhn@nist.gov
301-975-3337

OWASP AppSec DC

October 2005

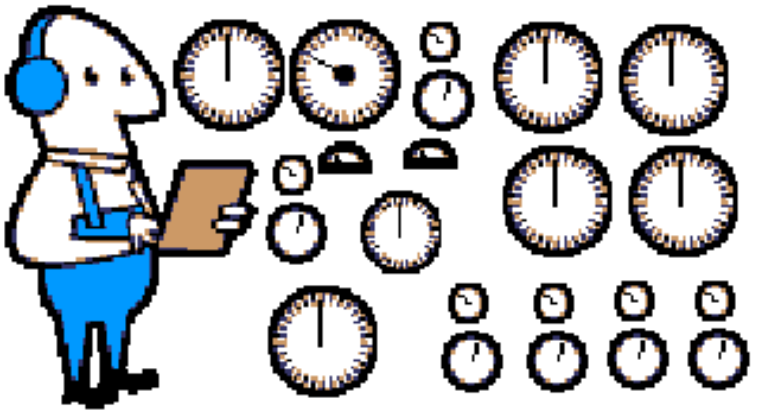
The OWASP Foundation
http://www.owasp.org/

Overview

- Goal - Determine if combinatorial testing ideas could be applied effectively, to reduce testing cost, because:
- If all faults triggered by a combination of *n* or fewer parameters, then testing all *n*-tuples of parameters can be considered pseudo-exhaustive testing, for some classes of software.
- Some findings are surprising, raise questions

Combinatorial Testing

- One approach to dealing with combinatorial explosion
- Consider: device with 20 inputs, 10 settings each
  - 10^{20} combinations
  - Which ones to test?



Combinatorial Testing Benefits

- Suppose no failure requires more than a pair of settings to trigger
- Then test all pairs – 180 test cases sufficient to detect any failure
- How many settings required in real-world software?
- If we know, can conduct “effectively exhaustive” testing

Combinatorial Testing Costs

- For k parameters with v values each number of test cases required for n-way interaction is proportional to $(v/2) \log_n k$ for small n
- Test combinations generated using algorithms for covering arrays

Effectiveness / coverage

- Reasonable testing goal: test all *n*-way combinations, where *n* is largest *n*-way interaction observed to cause failure in similar systems
- Questions
  - what is value of *n*?
  - does value differ for different types of software?
  - Is there a point of diminishing returns?

Empirical evidence - limited

- Dalal, et al., 1999 – effectiveness of pairwise testing, no higher degree interactions
- Smith, Feather, Muscetolla, 2000 – NASA Deep Space 1 software – pairwise testing detected 88% and 50% of flaws for 2 subsystems, no higher degree interactions
- Wallace, Kuhn, 2001 – medical device s/w – 98% of flaws were pairwise interactions, no failure required > 4 conditions to trigger
- Kuhn, Reilly, 2002 – browser and server software, no failure required > 6 conditions to trigger
- Kuhn, Wallace, Gallo, 2004 – NASA distributed scientific database software, no failure required > 4 conditions to trigger

Procedures

- Reviewed bug databases of two open source projects – Mozilla browser and Apache server, FDA recall reports, NASA development notes and bug reports
- Categorized reported bugs according to number of conditions required to trigger failure

Results

FTFI No.	Medical Devices	Browser	Server	NASA GSFC
1	66	28	41	67
2	97	76	70	93
3	99	95	89	98
4	100	97	96	100
5		99	96	
6		100	100	

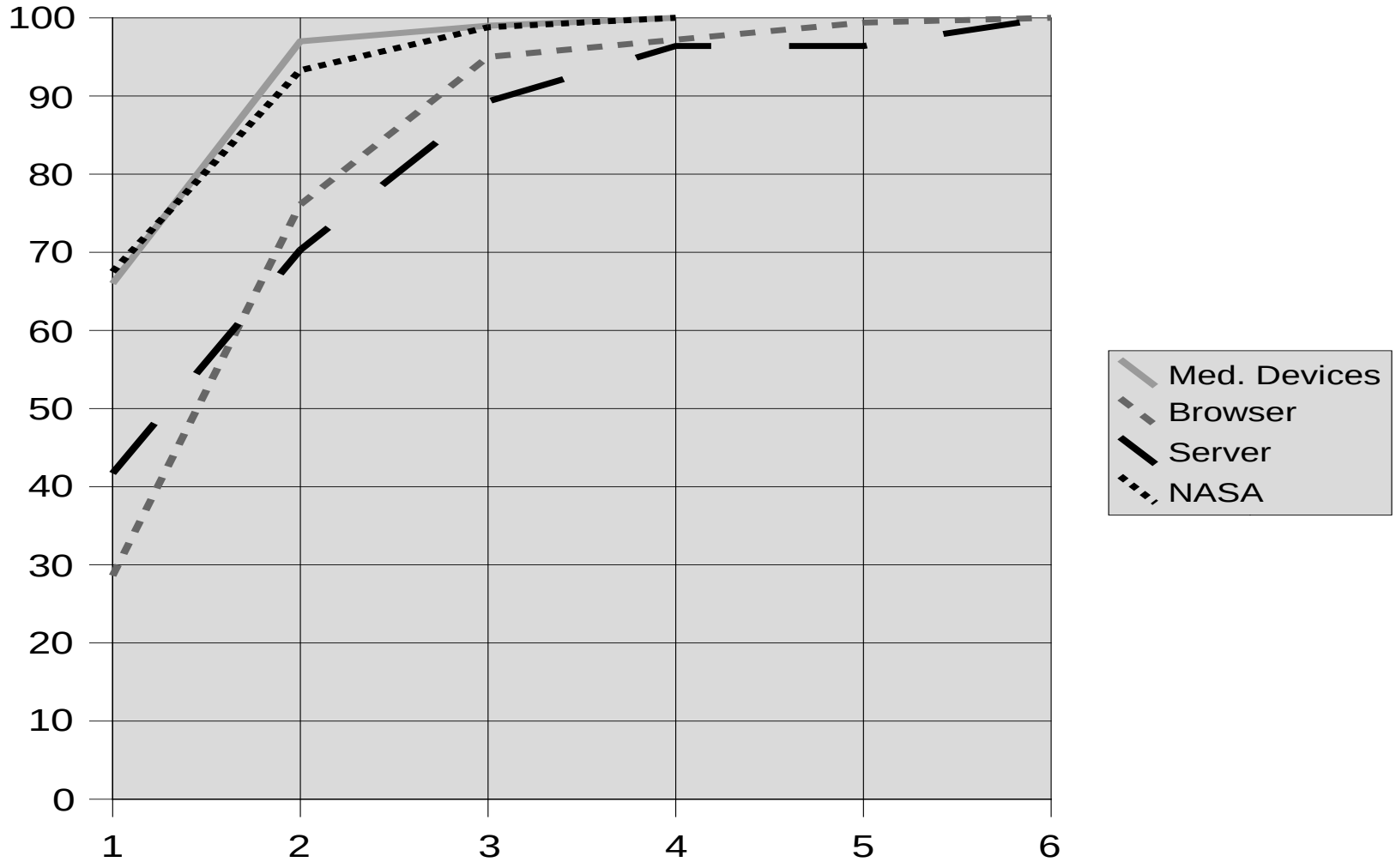
Other evidence

FTFI No.	RAX conver-gence	RAX correct-ness	RAX interface	RAX engine	POSIX modules	Medical Devices	Browser	Server	NASA GSFC
1	61	72	48	39	82	66	28	41	67
2	97	82	54	47	*	97	76	70	93
3	*	*	*	*	*	99	95	89	98
4	*	*	*	*	*	100	97	96	100
5	*	*	*	*	*		99	96	
6	*	*	*	*	*		100	100	

Additional findings

- Browser and server
  - pairwise testing would detect ~70%
  - 6-way testing would detect 100%
- Medical devices and NASA distributed database system
  - pairwise testing would detect >90%
  - 4-way testing would detect 100%
- These errors were less complex than browser and server errors!! Why?
  - More detailed reports?
  - Better testing (more eyes)?
  - Application characteristics?

Power law for failure triggering fault interactions?



Discussion

- Point of diminishing returns fairly low:
  - degree 2 interactions – 70% of bugs
  - degree 3 interactions – 90% of bugs
- Appropriate value of *n* may be 3 to 6
- Probably some “don’t care” conditions, so few or none may actually require > 4

Outlook

- Results imply that pseudo-exhaustive testing can be practical with automated test generation
- We need to know more about fault interactions in different application domains
- NIST is currently developing test tools based on these ideas – participation invited!
- Let me know if you are interested
- Rick Kuhn
kuhn@nist.gov or 301-975-3337

Papers

- D.Richard Kuhn, Dolores R. Wallace, Al J. Gallo, Jr., "Software Fault Interactions and Implications for Software Testing", *IEEE Trans. on Software Engineering,* vol. 30, no. 6, June, 2004).
- D.Richard Kuhn, Michael J. Reilly, "An Investigation of the Applicability of Design of Experiments to Software Testing", 27th NASA/IEEE Software Engineering Workshop, NASA Goddard Space Flight Center, 4-6 December, 2002.
- Dolores R. Wallace, D.Richard Kuhn, "Failure Modes in Medical Device Software: an Analysis of 15 Years of Recall Data ," *International Journal of Reliability, Quality, and Safety Engineering*, Vol. 8, No. 4, 2001