Judging the Value of Static Analysis

William Pugh
Univ. of Maryland

Judging static analysis

- Is static analysis an effective/profitable use of time/ funds/resources?
- Which static analysis tool or tools is most effective?
- How can I continuously improve the effectiveness of the use of static analysis?

- For all of these, want both a general answer, and a question about a particular situation.

What I want

- I want the field to move forward and improve
- I want software quality and security to improve

- I don’t particularly care about making life easier for some rule bound organization that can only do the right thing when the procedures for doing so are codified, filed in triplicate and filled out with a number 2 pencil.

Topics

- Benchmarks
- Secrecy
- Finding issues isn’t enough
- New issues
- Moving forward

Benchmarks

- Devise the benchmarks
- Measure the tools
- ... all set

Here be Dragons

Benchmarks are cursed

- Benchmarks always seem to be horrible code
- Vendors stop working on improving their tools and start working on improving their benchmark results
  - carried to extremes, spend huge amounts of effort on issues that rarely if ever occur in practice

Perfect club benchmarks

- Benchmarks for scientific applications, published in 1989
    - goal to see if compilers could effectively compile them for parallel computers
- Horrible code, nothing a human would ever want to maintain or modify

Perfect *Micro*benchmarks

- Not really application benchmarks
- Most programs had 1-3 important loop nests
  - containing 90+% of the CPU cycles
- Some grad student spend years getting their Ph.D. thesis developing techniques to parallelize one loop nest
  - no indication that the technique was generally useful

Perfect club benchmarks

- Were useful...
- For about 4 years, max.
- After that, they just distorted research in the field

SpecJVM98 DB benchmark

- The "database" benchmark
- 95+% cycles spent sorting a java.util.Vector
- Vector is synchronized, so papers were published showing how to determine that the Vector is thread confined so that you could remove the synchronization
  - ran 20-30+% speed improvement

Sorting???

- The db benchmark was a badly written hand coded shell sort
- Replace 20 lines of code with a one line call to the built in sort method
- program doubles in speed (100% faster)
  - even leaving in the synchronization

SpecJBB2000

- Java Business benchmark
- Ported from C++
- Rather than just construction objects
- each object construction performed 5 levels of method calls through Factory methods so that objects could be allocated near to other objects
  - meaningless in Java

SAMATE Java Reference Dataset

- 33 benchmarks
  - 3 don’t compile
  - average of 21 statements each

Test case 71: Switch fallthrough

- Yeah, the code is bad
- Really a code quality problem rather than a security issue
- I've got another 100 code quality patterns just as important that aren't being looked for

```
{
int month = 8;
switch (month) {
case 1: print("January");
case 2: print("February");
case 3: print("March");
case 4: print("April");
case 5: println("May");
case 6: print("June");
case 7: print("July");
case 8: print("August");
case 9: print("September");
case 10: print("October");
case 11: print("November");
case 12: print("December");
}
println(" is a great month");
}
```

Test 1754: Private Array-Typed field returned from a public method.

- no private field is returned from any method
- just a mistake in the benchmark
  - mistakes happen

```
public class PrivateArrayPublicMethod {
    private int[] foo = new int[1];

    public PrivateArrayPublicMethod() {
        foo = test(); /* BAD */
    }

    public int[] test() {

        int[] bar = new int[1];
        String inLine = null;
        int checkInput = ...;

        bar[0] = checkInput;
        return bar;
    }
```

Could be a problem...

- Fixed in JDK1.6.0-b61
- I'd been complaining about it for more than two years

```
public class JarEntry extends ZipEntry {

    Certificate[] certs;
    CodeSigner[] signers;

    public Certificate[]
                getCertificates() {
        return certs;
    }
    public CodeSigner[]
                getCodeSigners() {
        return signers;
    }
}
```

Sun’s JDK

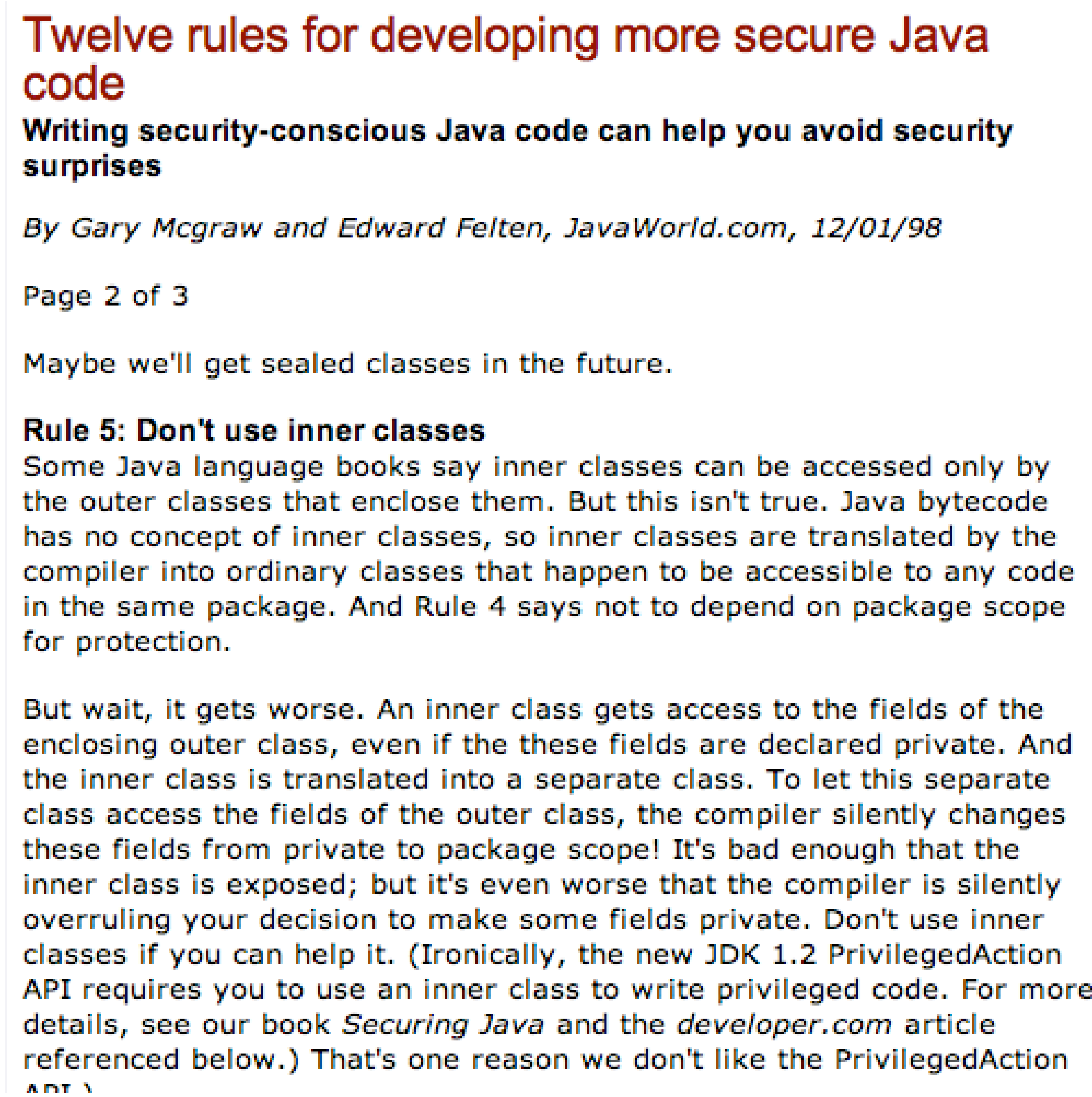
353	methods that return references to internal mutable structures
346	methods that store references to mutable structures without defensive copying
223	public static fields that can be modified by any untrusted applet

What is your threat model?

- Do you run with untrusted code in the same VM?
  - Of course, you don't have absolute trust in anything
- But is expending effort on this issue an effective use of your time?
  - code is made slower due to defensive copying

Are Inner Classes Evil?

- Some people also recommend against using inner classes because they can cause private members to be accessible to other code in the same package
    - stupid allocation of resources, in my opinion



Twelve rules for developing more secure Java code

Writing security-conscious Java code can help you avoid security surprises

By Gary Mcgraw and Edward Felten, JavaWorld.com, 12/01/98

Page 2 of 3

Maybe we'll get sealed classes in the future.

Rule 5: Don't use inner classes
Some Java language books say inner classes can be accessed only by the outer classes that enclose them. But this isn't true. Java bytecode has no concept of inner classes, so inner classes are translated by the compiler into ordinary classes that happen to be accessible to any code in the same package. And Rule 4 says not to depend on package scope for protection.

But wait, it gets worse. An inner class gets access to the fields of the enclosing outer class, even if the these fields are declared private. And the inner class is translated into a separate class. To let this separate class access the fields of the outer class, the compiler silently changes these fields from private to package scope! It's bad enough that the inner class is exposed; but it's even worse that the compiler is silently overruling your decision to make some fields private. Don't use inner classes if you can help it. (Ironically, the new JDK 1.2 PrivilegedAction API requires you to use an inner class to write privileged code. For more details, see our book *Securing Java* and the *developer.com* article referenced below.) That's one reason we don't like the PrivilegedAction

Opinions differ

- Just because Gary McGraw and Ed Felton say something doesn’t mean it is important
- Just because I say something something doesn’t mean it is important
- There are a lot of judgement calls that we shouldn’t bake into the qualification process

Hard coded password

- In the presence of the following complexities
  - scope, address alias level, container, local control flow, loop structure, buffer address type
- Really? All this is required of an acceptable static analysis tool?
  - Perhaps of some value, but I’m sure there are lots of things, not on the list, that are more important to check

SAMATE dataset isn’t bad

- Despite the issues I’ve raised, the SAMATE reference data set isn’t bad for a benchmark
  - most benchmarks are horrible
- But I’ve seen enough that concerns me that I don’t want it driving static analysis research for the next 10 years

Style vs. correctness

- Some things are always wrong
  - dereferencing a null pointer in C
  - SQL injection
- Some things are merely error prone
  - bad style

What matters?

- At Google, null pointer exceptions aren’t consider to be a serious problem in server code.
  - But at eBay, they are.
- Both eBay and Google have developed their own prioritized lists of which issues they care about
  - they are significantly different.

Problems with Benchmarks

- Quality issues
- Vendors optimize/innovate for the benchmarks, not the real problem
- Benchmarks outlive their usefulness
  - don’t evolve to match current issues
  - allow benchmark optimization to be taken to useless extremes

Benchmarks can serve as badnessometers

- If a tool fails miserably on a benchmark, the tool has some real problems
- But if a tool scores 99 or 100% on the benchmark, it doesn't give you confidence that the tool is effective
  - another tool with a score of 95% might be much more effective in practice

Secrecy

Everything is a trade secret

- What specific issues each tool finds (and which it doesn’t find)
  - general categories of issues are published
  - but not example results
- What experiences companies have with static analysis
- The secure coding standard and test cases used by Motorola (presented yesterday)

How bad is it?

- Coverity says that they no longer publish their results openly, because if they did "our competitors could improve their tools so they could report the same issues".
  - Coverity employees are forbidden to speak to me at academic conferences
- "Understanding the Value of Program Analysis Tools", published at OOPSLA, gives fictional numbers in reporting experience at eBay

Is this healthy?

- Initially, tools were finding easy stuff
    - knowing what to look for got you 95+% of the way to having a good tool
- I hope we've moved beyond that now
    - tools are more sophisticated (SAT solvers, etc)
    - knowing what issues you'd like to report doesn't give you a huge leg up

Can we require openness?

- Can we knock some heads?
  - People don't want to unilaterally disclose
- If you want compete for government contracts, you must publish your results on WebGoat and the SAMATE dataset
  - not just your score, but the actual way the data is presented to users?

Learning from others

- The field would advance more rapidly
- If researchers and companies could see what was effective, what worked well, what didn't work well

Usability

It isn't enough to just find issues

- The value of a tool depends on how effectively developers can use the tool to review and address issues
- A tools that simply dumps a text listing of issues to standard out is almost entirely useless
  - XML output is an optional feature?!?!?!

Tool Usability

- Code navigating in context of issues
- Understanding paths relevant to issue
  - particularly interprocedural paths
- Under what assumptions can this happen?
- Why should I care about this and how do I remedy it?
  - developer education is an important part of tool use

Issue management

- Can you see which issues are new
  - in the last commit/build?
  - since the last release?
- Can a team of 4 QA experts, 2 security wizards and 6 software developers effectively collaboratively audit an artifact?

Integration

- More and more companies are using more than one static analysis tool
  - does your tool play well with others?
- Integration into continuous build and integration system
- Integration into testing tools
  - show me the test cases that cover this line of code
- Can you tool be integrated into the companies bug tracking and version control system?

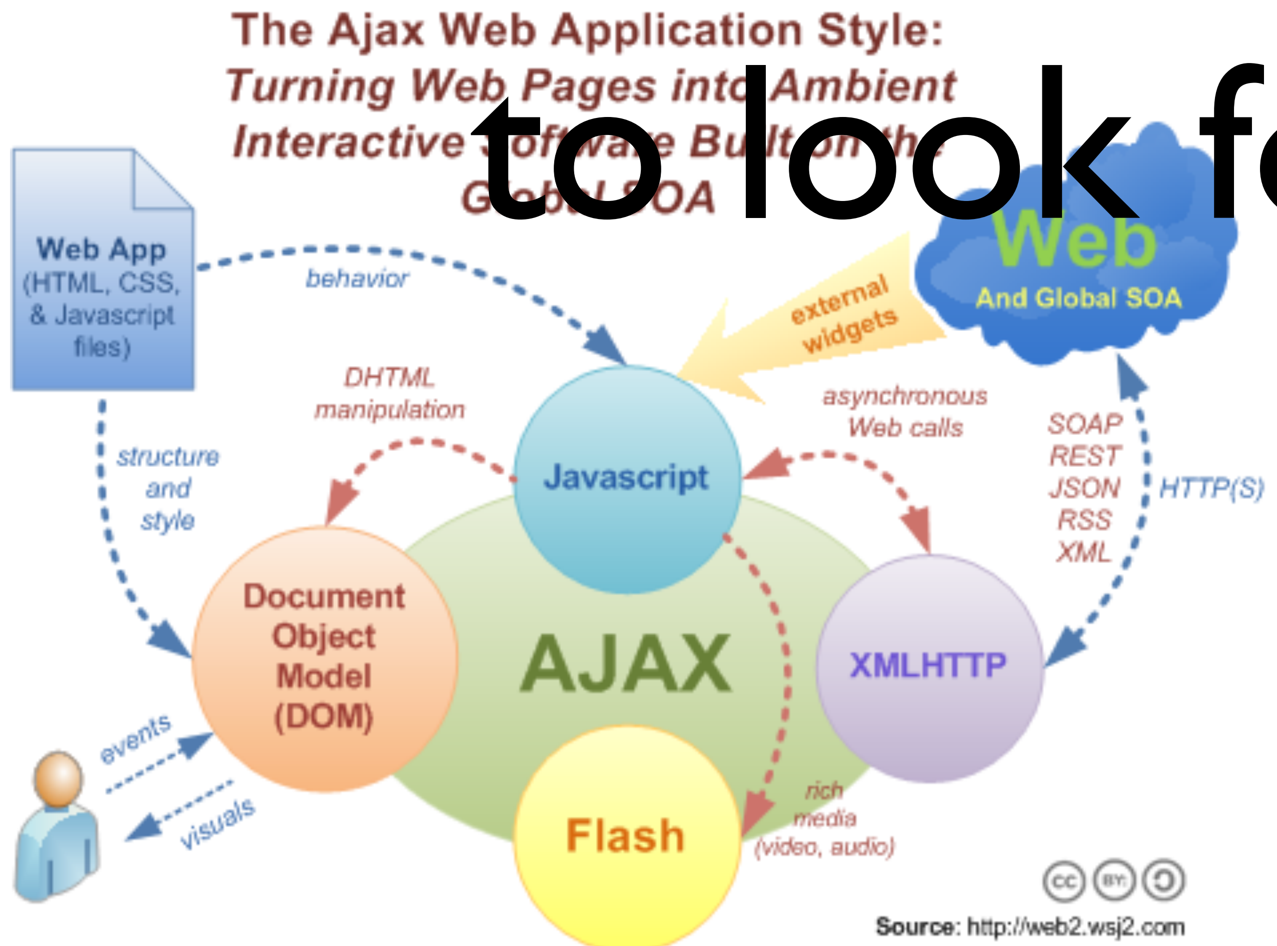
At Google

- Initial efforts had a QA team audit warnings, file bug reports when appropriate
  - Didn’t scale, didn’t get the reports to the right people at the right time.
- At Google, when you want to commit a change, someone else needs to review that change
  - static analysis is now part of that review process

Be great to study effectiveness

- But secrecy could doom us

We know what security vulnerabilities we need to look for, right?


The Ajax Web Application Style: Turning Web Pages into Ambient Interactive Software Built on the Global SOA
Web App (HTML, CSS, & Javascript files)
behavior
external widgets
Web
And Global SOA
DHTML manipulation
asynchronous Web calls
structure and style
Javascript
SOAP
REST
JSON
RSS
XML
HTTP(S)
Document Object Model (DOM)
AJAX
XMLHTTP
events
visuals
Flash
rich media (video, audio)
Source: http://web2.wsj2.com

You don’t want to be fixated

- Addressing buffer overflow, SQL injection and cross site scripting is a good thing
- But there is always something new to think about
- maybe for existing applications
- or maybe for entirely new applications

New issues in old applications

- In languages like C/C++, where anything that goes wrong can blow up your application
  - there is a nearly endless list of possible bugs to look for
    - e.g., negating a negative number can result in a negative number
  - They are there in Java too, they just don't completely blow up your application

Checking for design problems

- Does your web application have a logout link on each page?

New problem domains

- Cell phone applications (iPhone, gPhone, many more)
- AJAX
- AJAX frameworks (Google web toolkit, etc)
- Open Social

Moving forward

Benchmarks are good, but...

- be humble about their value
- Acing the benchmarks *doesn't* mean you are *good*
  - Flunking them means you have problems
  - Benchmarks are badness-ometers
- They need to be aged/retired to keep people from drilling down on aspects that aren't widely applicable

In my ideal world

- Emulate the Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, started in 1992.
- For each TREC, NIST provides a test set of documents and questions. Participants run their own retrieval systems on the data, and return to NIST a list of the retrieved top-ranked documents. NIST pools the individual results, judges the retrieved documents for correctness, and evaluates the results. The TREC cycle ends with a workshop that is a forum for participants to share their experiences.

How it would work

- Each year (maybe every other year), NIST would release sample applications for that competition
- Static analysis tool vendors would report their results
  - whatever they think is most significant
- Might also have teams that manually perform security audits
- Built in incentive for thinking outside the box, reporting problems no one else is looking for

Collect results

- Results would be collected, reviewed
- We'd learn something about the tools
- We'd learn something about the software that was reviewed
- We'd hopefully learn about new things that static analysis tools should be looking for

Still concerned about secrecy...

- The field needs controlled, independent studies of the *use* of commercial static analysis tools
  - Are they effective?
  - Which ways of using them is most effective?
- Not sure if this can or will ever happen.
- But wouldn't it be nice if static analysis tools competed on quality and effectiveness?

Discussion