SYNOPSYS®
Accelerating Innovation

Instrumentation in the era of Stack Map Frames

Peter Dillinger

18 Nov 2014

About me

- Studies in program analysis and verification
- 5 years at Coverity (acquired by Synopsys)
- Technical lead for Java analysis

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

About Coverity

- “Development testing”
  - Defects found with static analysis
    - “Quality” and “security”
  - Defects found with dynamic analysis
  - Testing gaps found with test analysis
  - Build and architecture analysis
- www.coverity.com
- http://scan.coverity.com
- http://code-spotter.com

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

About Dynamic Analyzer

- Finds:
  - Resource leaks
  - Deadlocks
  - Race conditions
  - … that might be too tricky for static analysis.
- Works via a Java agent doing bytecode instrumentation
- New in 7.6.0:
  - Agent injection
    - `cov-capture` *`test command`*
  - Java 8 support

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

About Dynamic Analyzer

- Finds:
  - Resource leaks
  - Deadlocks
  - Rac
  - … t
- Works
- New i
  - Age
    - co
  - Jav

harold seigel harold.seigel at oracle.com
Fri Mar 22 13:28:21 PDT 2013

Hi,

We have filed a CCC to deprecate -XX:-UseSplitVerifier in JDK 8 and it has been approved.

Previous users of -XX:-UseSplitVerify can initially use -noverify until
they have correct version 51 bytecodes. Also, due to requests from
folks who do generate version 51 bytecodes, we have added improvements
to the new verifier error diagnostics - and specifically targeted them
to give additional detailed information about stackmap tables. This
should help in the transition to generating correct version 51 class files.

Thanks, Harold

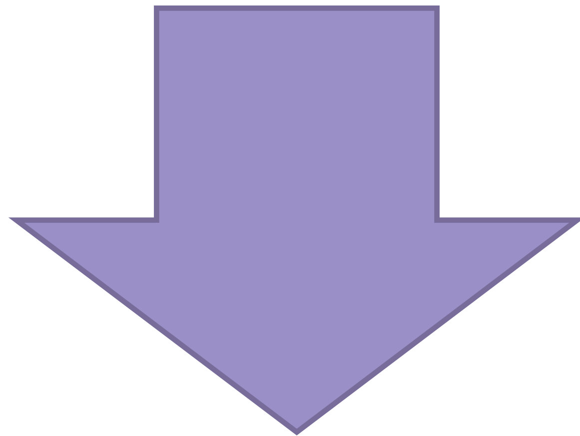
© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

The Story of Stack Maps

Thanks to Prashant Deva blog post

Bytecode verification is "too slow"



Bytecode requires types on operand stack

Java 5

Java 6 – experimental feature

Java 7 – default feature, with -XX:-UseSplitVerifier work-around

Java 8 – No work-around. Yet, JVM has old verifier!

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

Example

```
Comparator<String> getComparator(boolean b) {
    return b ? new C1() : new C2();
}
```

```
public java.util.Comparator<java.lang.String> get(boolean);
    Code:
       0: iload_1
       1: ifeq          14
       4: new           #18         // class C1
       7: dup
       8: invokespecial #20         // Method C1."<init>":()V
      11: goto          21
      14: new           #21         // class C2
      17: dup
      18: invokespecial #23         // Method C2."<init>":()V
      21: areturn
```

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

The big problem

- How ASM deals with it:
  - Option 1: Define stack maps yourself
  - Option 2: Let ASM compute them
- BUT… instrumentation agent
  - Referenced classes might not be loaded yet

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

Solutions

- Don't inject new control flow
  - Just insert straight line code
- Consider adding indirection / helper methods
- Implement a super smart instrumentation API

© 2014 Synopsys, Inc. All rights reserved.

SYNOPSYS® Accelerating Innovation

Peter Dillinger

www.coverity.com

SYNOPSYS®

Accelerating Innovation