DynJQual: Dynamic Checking of Type Qualifiers in Java

Ron Alford    Carlos Castillo    Sorelle Friedler

Department of Computer Science
University of Maryland, College Park

CMSC631 Project Presentation
Type Qualifiers

- Programmer defined types
- Annotates:
  - Fields
  - Variables
  - Function parameters and return values
foo(@untainted int);
@untainted int a = 54;
@tainted int b = 0;
if (c == 0) {
    b = a;
}
foo(b); // Error?
Dynamic Analysis

Type checking at runtime!

- **Pros:**
  - No false positives
  - Allows runtime defined types!

- **Cons:**
  - Can only find errors hit by test cases
  - Adds some performance overhead
DynJQual Class Transformation

```java
foo(@untainted int);
@untainted int a = 54;
@tainted int b = 0;
if (c == 0) {
    b = a;
}
foo(b);  // Error?
```

```java
foo(@untainted int);
@untainted int a = 54;
@tainted int b = 0;
if (c == 0) {
    b = a;
    SetAnnotation("b", "a");
}
foo(b);
VerifyAnnotation("foo", "b");
```
DynJQual: Dynamic Checking of Type Qualifiers in Java

- Transforming source code directly is unwieldy (sometimes impossible)
- Solution: Transform byte code directly
- New problem: How do we track annotations through function calls
The Shadow Stack

Instruction: +

Untainted
Tainted

...
The Shadow Stack (continued)

Instruction: +

\[ 7 + 11 = 18 \]

Untainted \( \cup \) Tainted = Tainted

\[
\begin{array}{c}
\hline
18 \\
1 \\
4 \\
\hline
\end{array}
\]

\[
\begin{array}{c}
\hline
\text{Tainted} \\
\vdots \\
\vdots \\
\hline
\end{array}
\]
Class project from Spring 2006

Initially supported much smaller segment of Java

Later work extended functionality to support arrays, standard library
Progress

- Ant build system
- Recompilation efficiency improvements ($\approx 15x$)
- Revamped byte-code recompilation scripts
- Ported to ASM 3.1 to help with the JSR/RET problems
Plans

- More bug fixes (JSR/RET, Array handling)
- Testing (and more bug hunting)
- Performance evaluation (SPEC)
- More applications?
Summary

- Dynamic type checking for Java
- Mostly implemented already
- TODO: Polish and run experiments