When Code Gets Spooky

Unveiling Hidden Errors Caused by Library Upgrades

Sebastian Böhm

Florian Sattler, Sven Apel

Chair of Software Engineering
Saarland University

March 25th, 2025

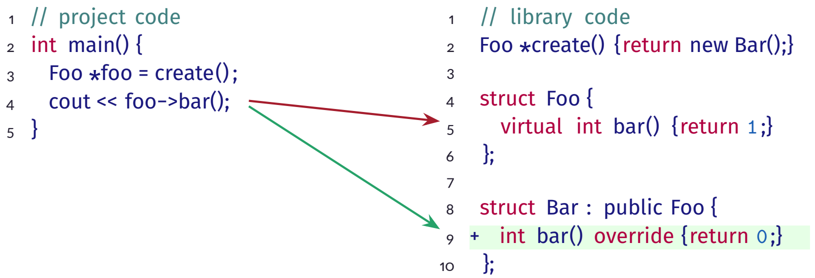
Someone Updates a Library...

```
commit f61346823615f5976ba68576cf6465076ee44bad
Author: Florian Sattler
Date: Wed Sep 27 09:55:52 2023 +0200

      Update phasar to current version.

phasar: v1.0.3 → v1.1.0
```

```
Testing Time: 1.98s
Passed: 229
Failed: 4
```

Why are the tests failing?

1.2 Example



```cpp
// project code
int main() {
  Foo *foo = create();
  cout << foo->bar();
}
```

```cpp
// library code
Foo *create() {return new Bar();}

struct Foo {
  virtual int bar() {return 1;}
};

struct Bar : public Foo {
+ int bar() override {return 0;}
};
```

A different function is called after update!

Example

// proje

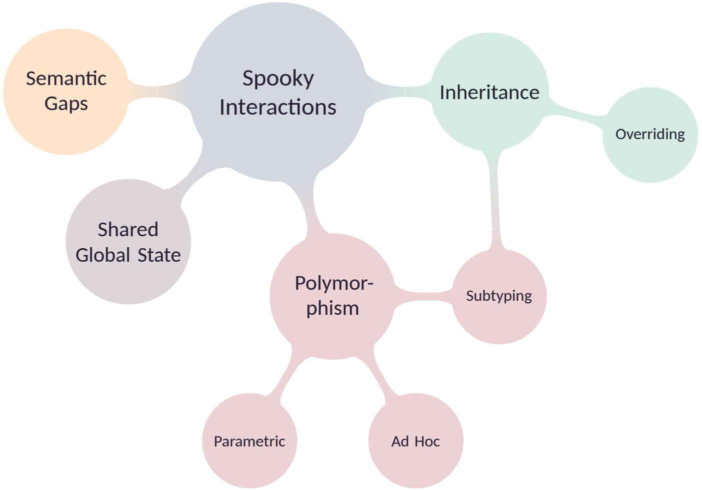
Spooky Interactions In Code

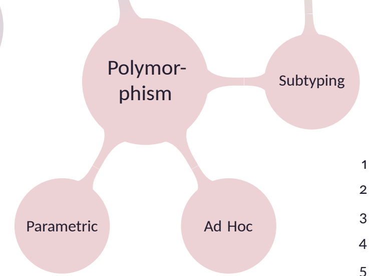
A *spooky interaction* is

- a code interaction
- changing without direct modification
- causing unexpected behavior

A different function is called after update!

What Causes Spooky Interactions?





```cpp
  template<typename T> T foo() {
    return 1;
  }

+ template<> int foo<int>() {
+   return 0;
+ }

  int main() {
    std::cout << foo<int>();
  }
```

Parametric polymorphism

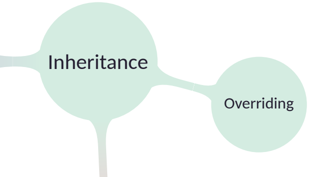
```cpp
  struct A {
    int i;
    A(int i) : i(i) {};
  };

  int foo(A a) { return a.i; }

+ int foo(int i) { return i - 1; }

  int main(int argc, char *argv[]) {
    std::cout << foo(argc);
  }
```

Ad-hoc polymorphism

```cpp
// project code
int main() {
  Foo *foo = create();
  cout << foo->bar();
}
```

```cpp
// library code
Foo *create() {return new Bar();}

struct Foo {
  virtual int bar() {return 1;}
};

struct Bar : public Foo {
+ int bar() override {return 0;}
};
```

Function Overriding



Shared Global State

```
  int global_n;
  int global_d = 1;

- void init(int n) {
+ void init(int n, int d = 0) {
    global_n = n;
+   global_d = d;
  }

  int divide() {
    return global_n / global_d
  }

  int main(int argc, char *argv[]) {
    init(42);
    std::cout << divide();
  }
```

Shared global state

```
  // returns a random int
  int randomInt();

  std::vector<int> getData(int n) {
    std::vector<int> data(n);
    std::generate(data.begin(), data.end(), randomInt);
-   std::sort(data.begin(), data.end());
    return data;
  }

  int main(int argc, char *argv[]) {
    auto data = getData(37);
    // expects data to be sorted
    processData(data);
  }
```

Semantic gap

Semantic Gaps

Why Are Spooky Interactions Problematic?

Software Design

- Do spooky interactions indicate technical debt?
- Which design decisions lead to spooky interactions?

Comprehension

- Is code with spooky interactions harder to understand?
- How can we help developers understand?

Security

- Do spooky interactions lead to security vulnerabilities? (e.g., rogue updates)

How can we detect spooky interactions?

How To Detect Spooky Interactions?

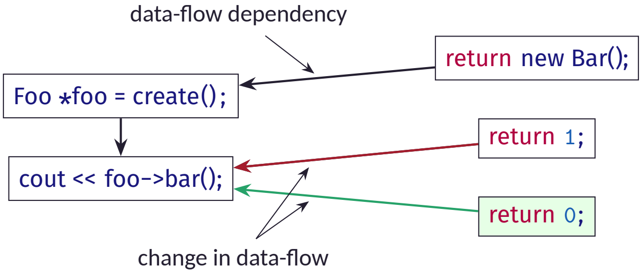
```
// project code
int main() {
  Foo *foo = create();
  cout << foo->bar();
}
```

```
// library code
Foo *create() {return new Bar();}

struct Foo {
  virtual int bar() {return 1;}
};

struct Bar : public Foo {
+ int bar() override {return 0;}
};
```

4.2

How To Detect Spooky Interactions?

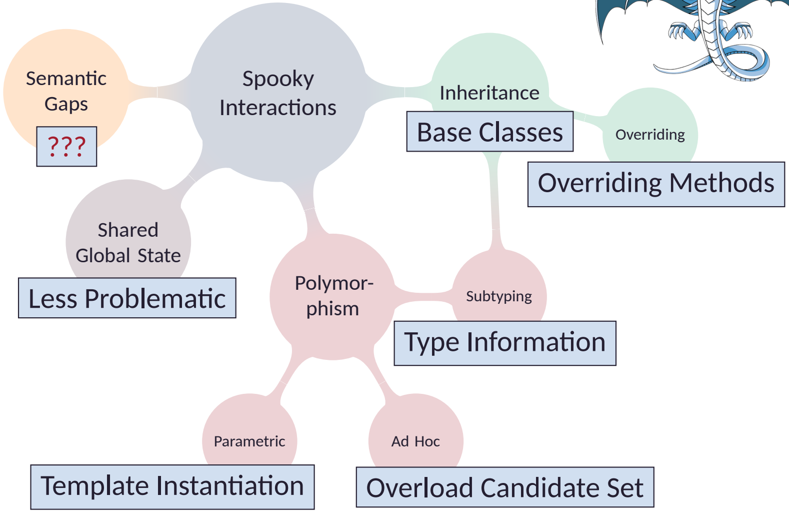
```
// project code
int main() {
  Foo *foo = create();
  cout << foo->bar();
}
```



```
// library code
Foo *create() {return new Bar();}

struct Foo {
  virtual int bar() {return 1;}
};

struct Bar : public Foo {
+  int bar() override {return 0;}
};
```

Issue: Too many false-positives!

Eliminating False-Positives



Sebastian Böhm

Florian Sattler, Sven Apel

Köthen, March 25th, 2025