Learning from 6,000 Projects

Lightweight Cross-Project Anomaly Detection

Natalie Gruska
Queen's University

Andrzej Wasylkowskii
Saarland University

Andreas Zeller
Saarland University
static int dcc_listen_init (...) {
  dcc->sok = socket(...);
  if (...) {
    while (...) {
      ... = bind (dcc->sok, ...);
    }
  }
  /* with a small port range, reUseAddr is needed */
  setsockopt (dcc->sok, ..., SO_REUSEADDR, ...);
} 
listen (dcc->sok, ...); 
}
static int dcc_listen_init (...) {
    dcc->sok = socket(...);
    if (...) {
        while (...) {
            ... = bind (dcc->sok, ...);
        }
    }
    /* with a small port range, reUseAddr is needed */
    setsockopt (dcc->sok, ..., SO_REUSEADDR, ...);
} 
listen (dcc->sok, ...);

bind < listen
setsockopt < listen
setsockopt < bind

should be called before bind()
Anomaly Detection
Anomaly Detection
Anomaly Detection

We need more examples!
Cross-project Anomaly Detection

Knowledge base

bind < listen
setsockopt < listen
setsockopt < bind

Program

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind
Cross-project Anomaly Detection

Knowledge base

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

bind < listen
setsockopt < listen
setsockopt < bind

Program

bind < listen
setsockopt < listen
setsockopt < bind
Cross-project Anomaly Detection

- Goal: Learn from thousands of other projects
Lightweight Parser: Focus Languages

Java   C++   C   PHP   Javascript

similar syntax:

{...} ;  foo()

similar keywords:

while   if   switch   return
void foo () {
    int fA;
    int fB = open("newFile");
    fA = open("myFile");
    while(j > 3){
        read(fA);
        write(fB, "Hello");
    }
    close(fA);
    close(fB);
}
void foo () {
    int fA;
    int fB = open("newFile");
    fA = open("myFile");
    while(j > 3){
        read(fA);
        write(fB, "Hello");
    }
    close(fA);
    close(fB);
}
fB = open(CONST)
fA = open(CONST)

Loop:
    read(fA)
    write(fB, CONST)

close(fA)

close(fB)
Loop:
read(fA)
write(fB, CONST)
close(fA)
close(fB)
**Source Code**

- `fB = open(...)`
- `fA = open(...)`
- `read(fA)`
- `write(fB, ...)`
- `close(fA)`
- `close(fB)`

**Abstract Representation**

**Function Models**

**Temporal Properties**
Abstract Representation

Function Models

Temporal Properties

fB: open < write
write < write
write < close
open < close

fA:
open < read
read < read
read < close
open < close
**Source Code**

1. `fB = open(...)`
2. `fA = open(...)`
3. `read(fA)`
4. `write(fB,...)`
5. `close(fA)`
6. `close(fB)`

**Temporal Properties**

- `open < write`
- `open < read`
- `write < write`
- `read < read`
- `write < close`
- `read < close`
- `open < close`
void foo () {
    int fA;
    int fB = open("newFile");
    fA = open("myFile");
    while(j > 3){
        read(fA);
        write(fB, "Hello");
    }
    close(fA);
    close(fB);
}
Knowledge Base

- f1 < close
- foo < bar
- open < write
- write < close
- lock < unlock
- malloc < free
The Knowledge Base

- Gentoo Linux distribution
- C projects
  - ≈6,000 projects
  - ≈200,000,000 lines of code
  - ≈16,000,000 temporal properties
- Creation time: 18h (11s per project)
Using a Knowledge Base

Knowledge Base

P1
f1
f2

P2
f1
f2
f3
Using a Knowledge Base

Knowledge
Base

Program

P1

f1

f2

P2

f2

f3

noths < inet_ntoa
memset < sizeof
bind < listen
setsockopt < listen
Using a Knowledge Base

Knowledge Base

Program

f1

f2

f3

P1

P2

noths < inet_ntoa
memset < sizeof
bind < listen
setsockopt < listen
Finding Patterns

Program's functions

<table>
<thead>
<tr>
<th>Functions</th>
<th>Temporal Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>a&lt;b c&lt;d a&lt;e c&lt;a ...</td>
</tr>
<tr>
<td>f2</td>
<td></td>
</tr>
<tr>
<td>P1.f1</td>
<td></td>
</tr>
<tr>
<td>P1.f2</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Functions in the knowledge base
Finding Patterns

Program’s functions

Temporal Properties

\begin{align*}
a &< b \\
c &< d \\
a &< e \\
c &< a \\
\ldots
\end{align*}

Functions in the knowledge base

f1

f2

P1.f1

P1.f2

\ldots

\ldots

This is a pattern
Detecting Violations

Program's functions

<table>
<thead>
<tr>
<th>Temporal Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>a&lt;b</td>
</tr>
</tbody>
</table>

Functions in the knowledge base

f1
f2
P1.f1
P1.f2
...

Functions in the knowledge base

...
## Detecting Violations

Program's functions:

<table>
<thead>
<tr>
<th>Functions</th>
<th>P1.f1</th>
<th>P1.f2</th>
<th>f1</th>
<th>f2</th>
</tr>
</thead>
</table>

Temporal Properties:

- $a < b$
- $c < d$
- $a < e$
- $c < a$
- ...

This is a violation!
Evaluation

- 20 randomly chosen projects
- Ran anomaly detection on each of them
  - Classify top 25%
- Defects, code smells, false positives
20 Projects

- Between 69 and 595,664 SLOC (C only) (generated using David A. Wheeler's 'SLOCCount')
- 136 violations found in 11 projects
- Analysis time per project < 6 minutes (gimp < 18 minutes)
static int dcc_listen_init (…) {
    dcc->sok = socket(…);
    if (…) {
        while (…) {
            … = bind (dcc->sok, …);
        }
    }
    /* with a small port range, reUseAddr is needed */
    setsockopt (dcc->sok, …, SO_REUSEADDR, …);
}
listen (dcc->sok, …);

should be called before bind()
Defect in cksfv-1.3.13

```c
static int find_file (...) {
    DIR *dirp;
    struct dirent *dirinfo;
    ...
    dirp = opendir(".");
    if (dirp == NULL) {
        ...
    }
    while ((dirinfo = readdir(dirp)) != NULL) {
        ...
    }
    rewinddir(dirp);
    return 1;   \textcolor{red}{\textbf{should call} \textit{closedir}()} \textbf{instead}
}
```
Code smell in gimp-2.6.6

static gboolean gimp_page_selector_item_width_idle (...) {
    GimpPageSelectorPrivate *priv = ...;
   GtkTreeModel *model = GTK_TREE_MODEL (priv->store);
    GtkTreeIter iter;
    ...
    for (... = gtk_tree_model_get_iter_first (model, &iter);
         ...;
         ... = gtk_tree_model_iter_next (model, &iter))
    {
        ...
        gtk_tree_model_get (GTK_TREE_MODEL (priv->store),
                            &iter, ...,
                            ...);
        ...
    }
    ...
}
Global true positive rate: 22%
How to Improve Things?

- Use **CTL formulas** instead of temporal properties
- Explore **API evolution**
- Take **user feedback** into account
Your result is now ready. Please click on a violation (red marked line) to see a detailed description for the violation.

A call to closedir() is probably missing. There is a potential problem with a call to opendir().

return value of opendir()

1st arg of readdir()  1st arg of closedir()

LEGEND
- - - Missing data flow  - - Data flow in your code

>> Textual representation

DATA FLOW IN YOUR CODE:
- 1st arg of readdir() -> 1st arg of readdir()
- return value of opendir() -> 1st arg of readdir()

MISSING DATA FLOW:
- return value of opendir() -> 1st arg of closedir()

>> Example code for valid data flow

- function "find_search_dir", line 54
- function 'findpost_search', line 137
- function 'find_timebase', line 29

3
DIR *dirp;
struct dirent *dirinfo;
Cross-project Anomaly Detection

- Goal: Learn from thousands of other projects

Using a Knowledge Base

void foo () {
    int fa;
    int fb = open("newFile");
    fa = open("myFile");
    while(j > 3){
        read(fa);
        write(fb, "Hello");
    }
    close(fa);
    close(fb);
}

open < write
write < write
read < read
write < close
read < close
open < close

Summary