Gulfstream

Staged Static Analysis for Streaming JavaScript Applications

Salvatore Guarnieri

University of Washington

Ben Livshits

Microsoft Research
Web application

Third Party Server

Web page

widget.js
Safe Code Inclusion In JavaScript

Runtime Enforcement
• Conscript [Oakland 10]
• BrowserShield [OSDI 06]
• Caja

Static Analysis
• Gatekeeper [USENIX Sec 09]
• Staged Information flow for JavaScript [PLDI 09]

Whole program analysis approaches require the entire program
<table>
<thead>
<tr>
<th>Request</th>
<th>Status Code</th>
<th>Response Size</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET 3vyds43ujs</td>
<td>200 OK</td>
<td>10.2 KB</td>
<td>111 ms</td>
</tr>
<tr>
<td>GET 7i3rymig.js</td>
<td>200 OK</td>
<td>2.9 KB</td>
<td>33 ms</td>
</tr>
<tr>
<td>GET 8ldr4tl.js</td>
<td>200 OK</td>
<td>3.5 KB</td>
<td>53 ms</td>
</tr>
<tr>
<td>GET 4yghkao0ljs</td>
<td>200 OK</td>
<td>8.2 KB</td>
<td>4.2 ms</td>
</tr>
<tr>
<td>GET 1r8zakvz.js</td>
<td>200 OK</td>
<td>31.3 KB</td>
<td>128 ms</td>
</tr>
<tr>
<td>GET 4xotikfvs.js</td>
<td>200 OK</td>
<td>13.7 KB</td>
<td>6.5 ms</td>
</tr>
<tr>
<td>GET 76v00gik.js</td>
<td>200 OK</td>
<td>9.6 KB</td>
<td>79 ms</td>
</tr>
<tr>
<td>GET 88kp44c.js</td>
<td>200 OK</td>
<td>5.4 KB</td>
<td>79 ms</td>
</tr>
<tr>
<td>GET 43kaxl04.js</td>
<td>200 OK</td>
<td>530 B</td>
<td>17 ms</td>
</tr>
<tr>
<td>GET 97drnbow.js</td>
<td>200 OK</td>
<td>11 KB</td>
<td>17 ms</td>
</tr>
<tr>
<td>GET 344u3oqg.js</td>
<td>200 OK</td>
<td>7.5 KB</td>
<td>27 ms</td>
</tr>
<tr>
<td>GET 1hrw70k.js</td>
<td>200 OK</td>
<td>4.2 KB</td>
<td>27 ms</td>
</tr>
<tr>
<td>GET 7c3qom6e.js</td>
<td>200 OK</td>
<td>537 B</td>
<td>17 ms</td>
</tr>
<tr>
<td>GET 7q88khxq.js</td>
<td>200 OK</td>
<td>622 B</td>
<td>17 ms</td>
</tr>
<tr>
<td>GET 5vyds43ujs</td>
<td>200 OK</td>
<td>1.6 KB</td>
<td>17 ms</td>
</tr>
<tr>
<td>GET 1rlyly26.js</td>
<td>200 OK</td>
<td>2 KB</td>
<td>17 ms</td>
</tr>
</tbody>
</table>

17 requests, 122 KB, 1.11s (onload: 1.7s)
JavaScript programs are streaming
Script Creation

What does f refer to?

```html
<HTML>
  <HEAD>
  <SCRIPT>
    function foo(){...}
    var f = foo;
  </SCRIPT>
  <SCRIPT>
    function bar(){...}
    if (...) f = bar;
  </SCRIPT>
  </HEAD>
  <BODY onclick="f();"> ...</BODY>
</HTML>
```
Incremental Loading in Facebook

- Profile: 13 KB
- Inbox: 20 KB
- Friends: 29 KB
- Home: 157 KB

Total: 199 KB, 71% of total
Gulfstream In Action

Offline

Online
Gulfstream In Action

Offline

Online
Gulfstream In Action

Offline

Online
Outline

• Motivation
• Implementation
• Evaluation
• Conclusions
Queries

• We want to determine something about the program

• Example
  – What does f() refer to
  – Detect alert calls
  – Does this program use setTimeout
Points-To Analysis

• Provides deep program understanding

• Can be used to construct call graphs

• Is the foundation of further analyses

• Answers a simple question: What heap locations does variable $x$ point to
Points-To Example

1. var A = new Object();
2. var B = new Object();
3. x = new Object();
4. x.foo = new Object();
5. y = new Object();
6. y.bar = x;
7. y.add = function(a, b) {}
8. y.add(A, B)
Implementation Strategies

Datalog with bddbddb
+ Fast for large programs
+ Highly tuned
- Large startup cost
- Difficult to implement in the browser

• Used in Gatekeeper [USENIX Sec 09]

Graph-based flow analysis
+ Very small startup cost
+ Customized to work with Gulfstream
- Does not scale well
Implementation

• Normalize JavaScript
  – Turn program into a series of simple statements
  – Introduce temporaries as necessary

• Create flow graph – Use normalized program to generate flow constraints

• Serialize flow graph – Encode the flow-graph so online analysis can use it to update results
Implementation Continued

• Perform points-to analysis
  – Traverse flow graph to find all aliases
  – Follow flow through method boundaries
  – Generate points-to map for queries to use

• Queries – Use points-to data and flow graph to answer queries
Evaluation

• Question – Is Gulfstream faster than non-staged analysis

• Benchmarks
  – Synthetically generated
  – Scraped from Google code
  – Scraped from Facebook

• Simulate diverse environments
  – CPU speed and network properties
  – Cell phone, laptop, desktop, etc.
After 30KB of updates, Gulfstream is no longer faster.
Simulated Devices

- Low power mobile
- High power
<table>
<thead>
<tr>
<th>ID</th>
<th>Configuration Name</th>
<th>CPU coef. $c$</th>
<th>Link type</th>
<th>Latency $L$ in ms</th>
<th>Bandwidth $B$ in kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>G1</td>
<td>67.0</td>
<td>EDGE</td>
<td>500</td>
<td>2.5</td>
</tr>
<tr>
<td>2</td>
<td>Palm Pre</td>
<td>36.0</td>
<td>Slow 3G</td>
<td>500</td>
<td>3.75</td>
</tr>
<tr>
<td>3</td>
<td>iPhone 3G</td>
<td>36.0</td>
<td>Fast 3G</td>
<td>300</td>
<td>12.5</td>
</tr>
<tr>
<td>4</td>
<td>iPhone 3GS 3G</td>
<td>15.0</td>
<td>Slow 3G</td>
<td>500</td>
<td>3.75</td>
</tr>
<tr>
<td>5</td>
<td>iPhone 3GS WiFi</td>
<td>15.0</td>
<td>Fast WiFi</td>
<td>10</td>
<td>75.0</td>
</tr>
<tr>
<td>6</td>
<td>MacBook Pro 3G</td>
<td>1</td>
<td>Slow 3G</td>
<td>500</td>
<td>3.75</td>
</tr>
<tr>
<td>7</td>
<td>MacBook Pro WiFi</td>
<td>1</td>
<td>Slow WiFi</td>
<td>100</td>
<td>12.5</td>
</tr>
<tr>
<td>8</td>
<td>Netbook</td>
<td>2.0</td>
<td>Fast 3G</td>
<td>300</td>
<td>12.5</td>
</tr>
<tr>
<td>9</td>
<td>Desktop WiFi</td>
<td>0.8</td>
<td>Slow WiFi</td>
<td>100</td>
<td>12.5</td>
</tr>
<tr>
<td>10</td>
<td>Desktop T1</td>
<td>0.8</td>
<td>T1</td>
<td>5</td>
<td>1,250.0</td>
</tr>
</tbody>
</table>
Lessons Learned

• Slow devices benefit from Gulfstream

• A slow network can negate the benefits of the staged analysis

• Large page updates don’t benefit from Gulfstream
Facebook Experiment

• Visit 4 pages
  – Home
  – Friends
  – Inbox
  – Profile

• Each page loads additional JavaScript
Gulfstream Savings: Slow Devices
Gulfstream Savings: Fast Devices

10 seconds saved
Conclusion

• Gulfstream, staged analysis for JavaScript

• Staged analysis
  – Offline on the server
  – Online in the browser

• Wide range of experiments
  – For small updates, Gulfstream is faster
  – Devices with slow CPU benefit most
The End

• Contact: salvatore.guarnieri@gmail.com