Clang/LLVM for Automated Defense

John Criswell
University of Illinois at Urbana-Champaign
Automated Defense

A compiler that automatically generates executables that are protected against one or more broad classes of attacks.
Trouble on the Horizon

- We *depend* on **vulnerable** computing systems
- Attackers are better funded and more motivated
- We can’t fix all the bugs
- Automated defense can help defend us
Clang Has Automated Defenses

- Stack Canaries are bypassed by students for homework
- Partial bounds checks aren’t comprehensive
Potential Ways Forward

<table>
<thead>
<tr>
<th>System</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Flow Integrity (CFI)</td>
<td>8%</td>
</tr>
<tr>
<td>Write Integrity Testing (WIT)</td>
<td>18%</td>
</tr>
<tr>
<td>SAFECODE (PLDI 2006)</td>
<td>30%</td>
</tr>
<tr>
<td>Baggy Bounds Checking</td>
<td>68%</td>
</tr>
<tr>
<td>SoftBound + CETS</td>
<td>116%</td>
</tr>
</tbody>
</table>

Note: Table is **not** comprehensive.
From Research to Practice

• Stop trying to use weak solutions
• Use best of breed techniques
• Realistic discussions of security/performance tradeoffs
• Build what people need