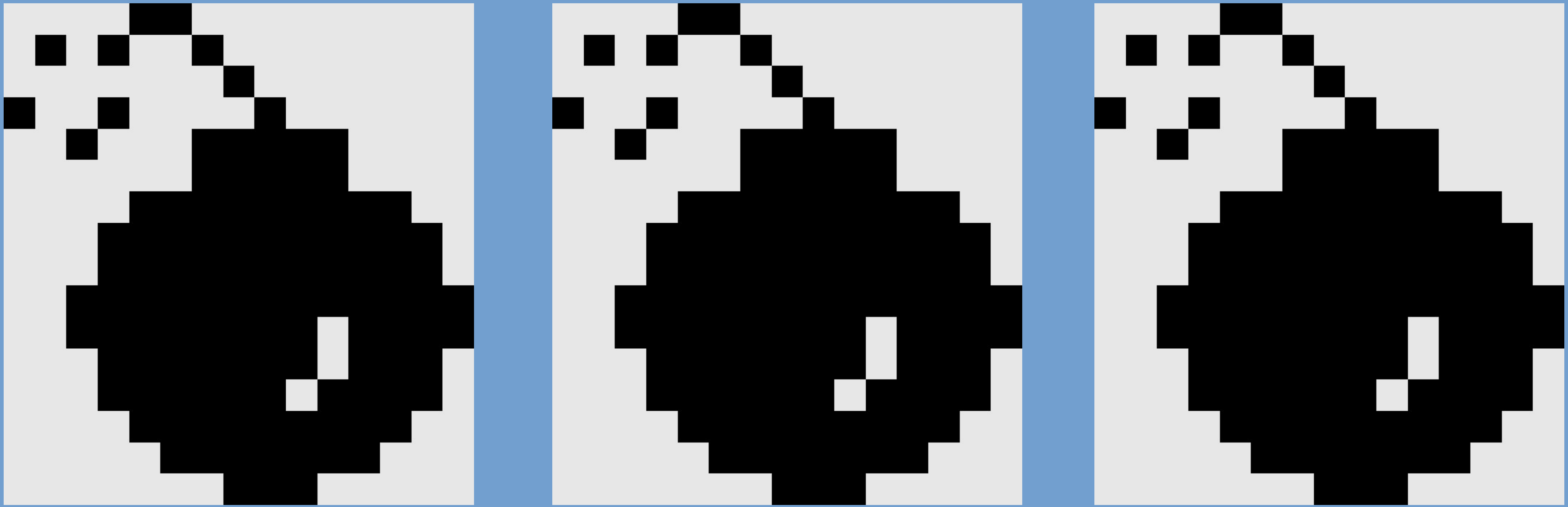
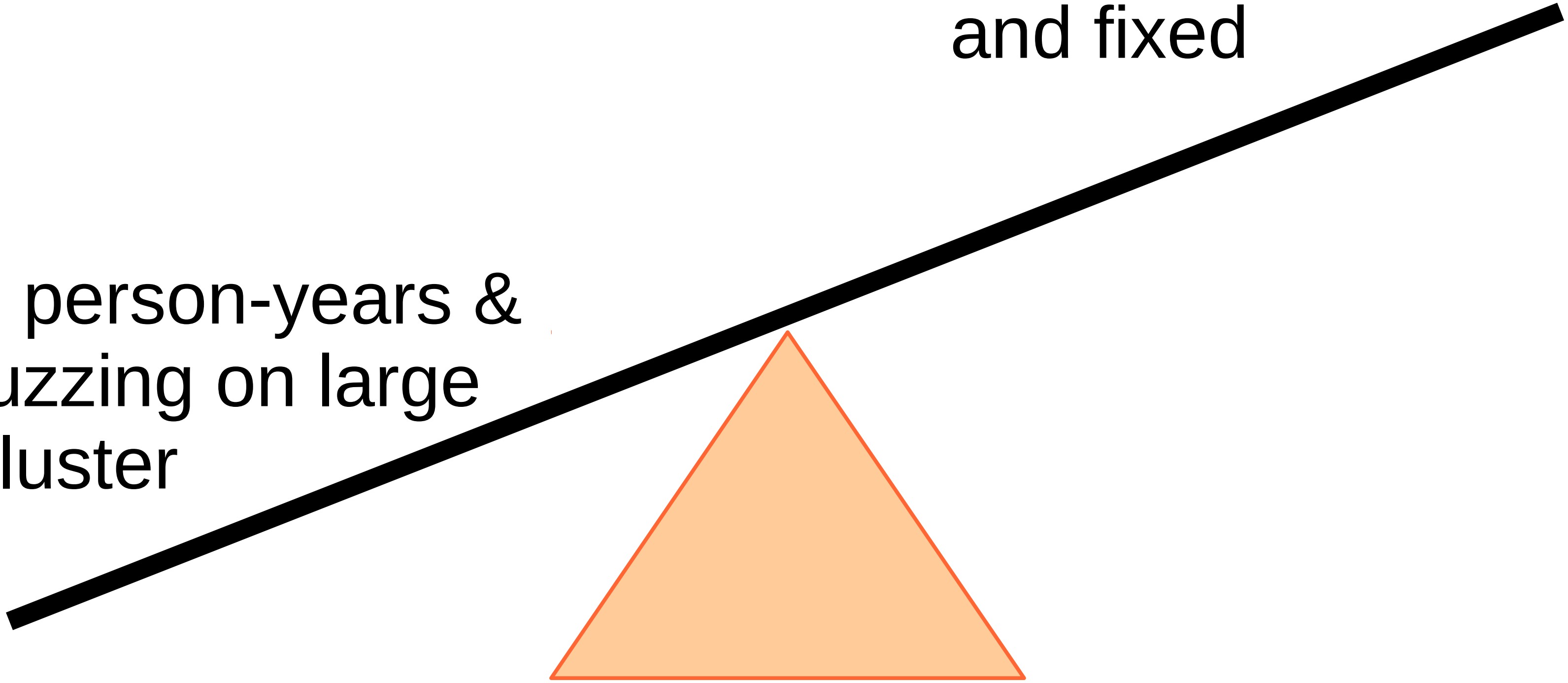


WarGames in memory:

Protecting applications in the presence of bugs

Mathias Payer <mpayer@purdue.edu>
Assistant Professor, Purdue University

FFmpeg and a thousand fixes



>1,000 bugs found and fixed

2 person-years & fuzzing on large cluster

http://j00ru.vexillium.org/?p=2211 Jan-10, 2014

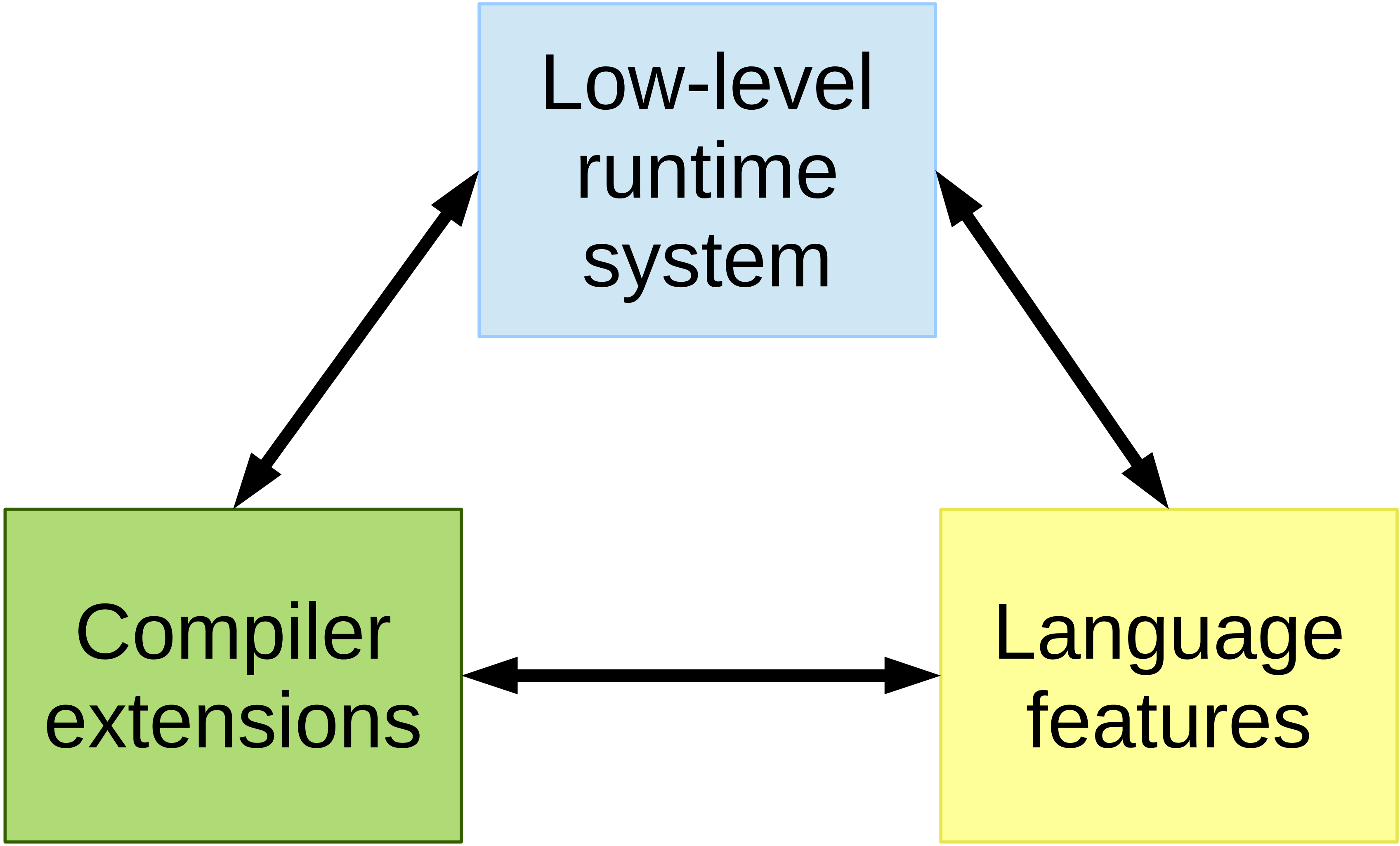
Software is unsafe and insecure

- Low-level languages (C/C++) trade type safety and memory safety for performance
  - Programmer responsible for all checks

- Large set of legacy and new applications written in C / C++ prone to memory bugs

- Too many bugs to find and fix manually
  - Protect integrity through safe runtime system

Detect, protect, defend

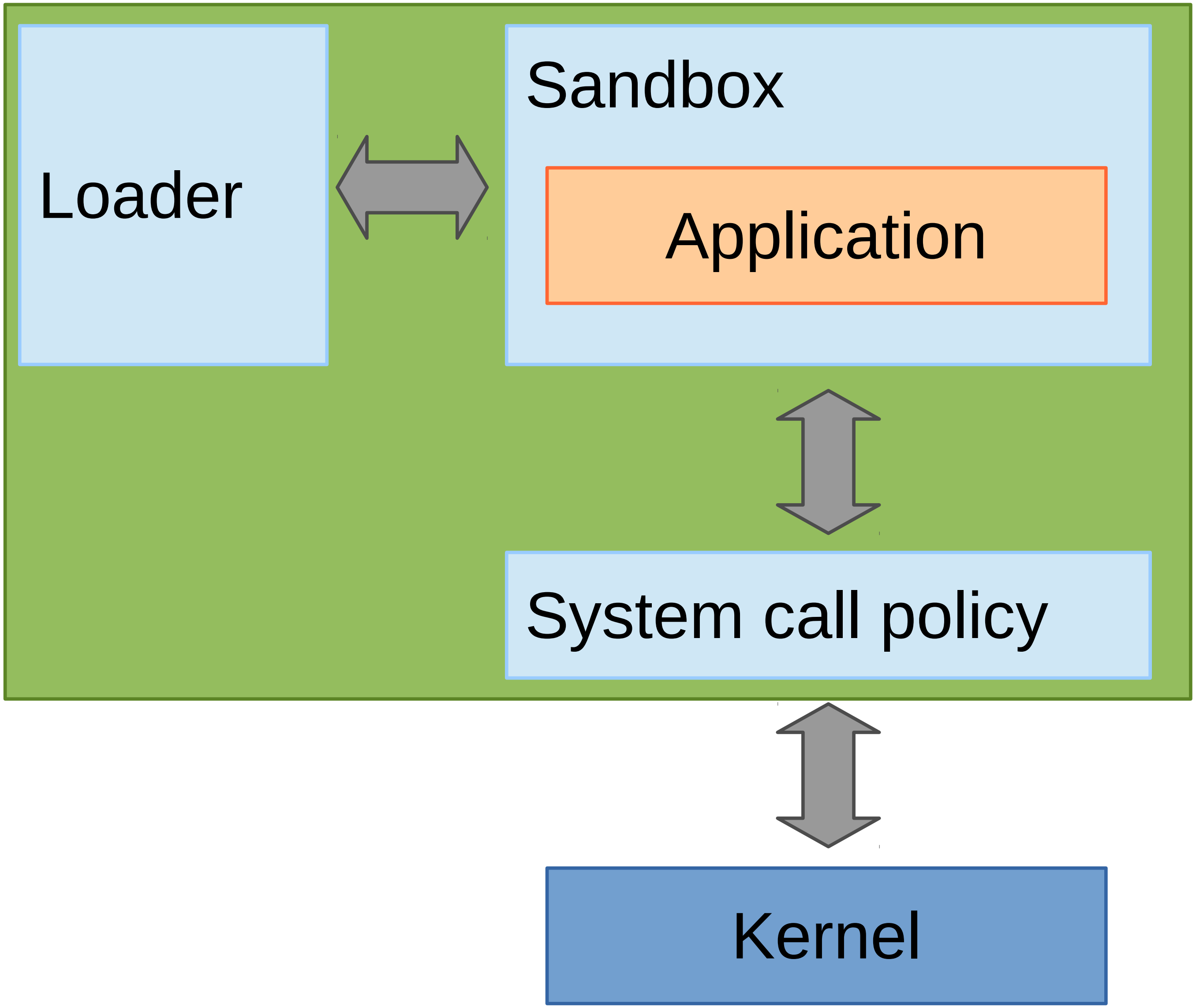


Low-level runtime system

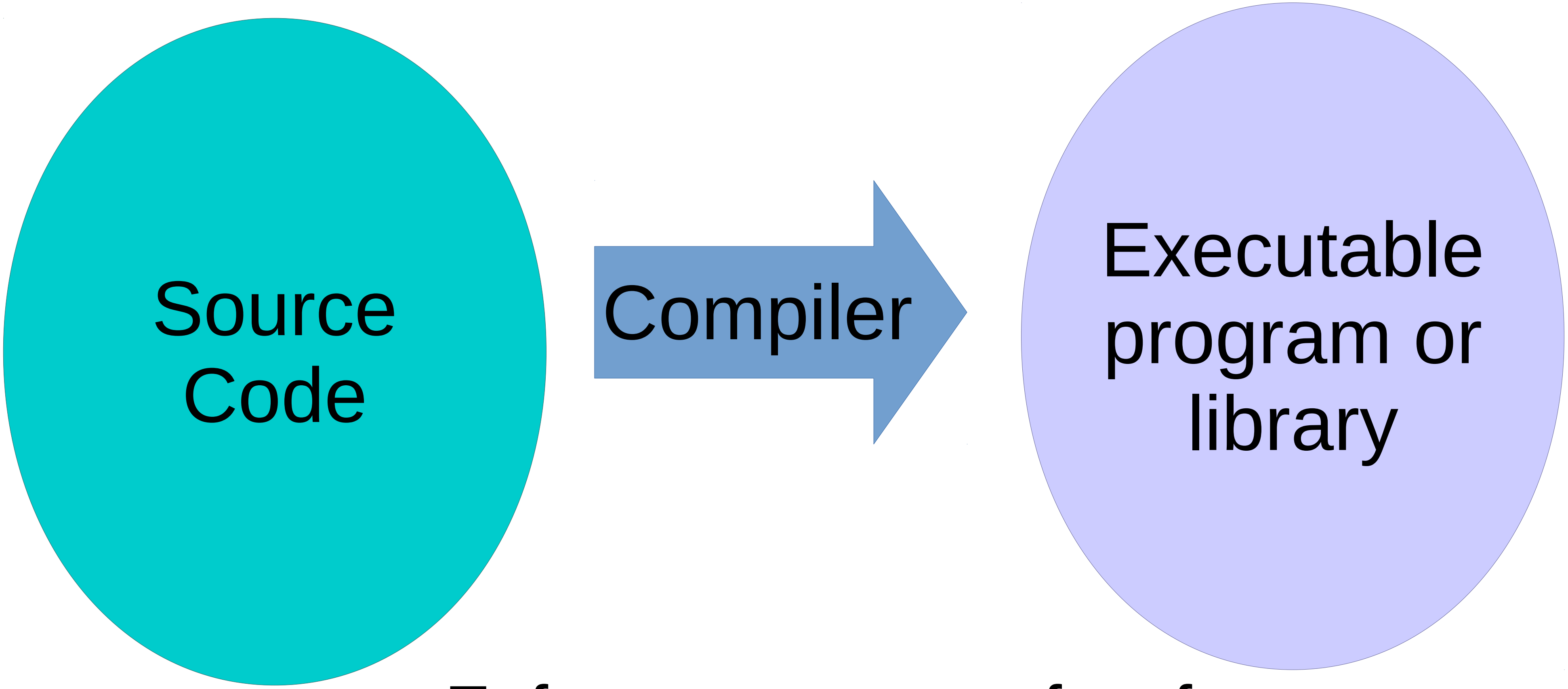
Compiler extensions

Language features

Low-level runtime system



Compiler extensions



- Enforce memory safety for a subset of data
- Embed high level details, enforce runtime protection

Conclusion

- Protect applications in the presence of bugs
  - Assume that unpatched vulnerabilities exist
- Enforcing strong policies for code
  - For existing binaries, source code and language extensions, and new languages

Mathias Payer <mpayer@purdue.edu>
Assistant Professor, Purdue University