Characterizing Configuration Complexity in Highly-Configurable Systems with Variational Call Graphs

Gabriel Ferreira, Christian Kästner, Jürgen Pfeffer, Sven Apel

Configuration options create challenges for security

- Human analysis capacity does not scale with exponentially growing configuration space
- Interactions among configuration options may lead to surprising effects and vulnerabilities
- Vulnerabilities may hide in specific configuration combinations
- Challenging to verify security properties in all combinations
- Developers maximize functionality to configuration complexity
- Current approaches:
  Work on specific configurations
  Do not explore configuration knowledge

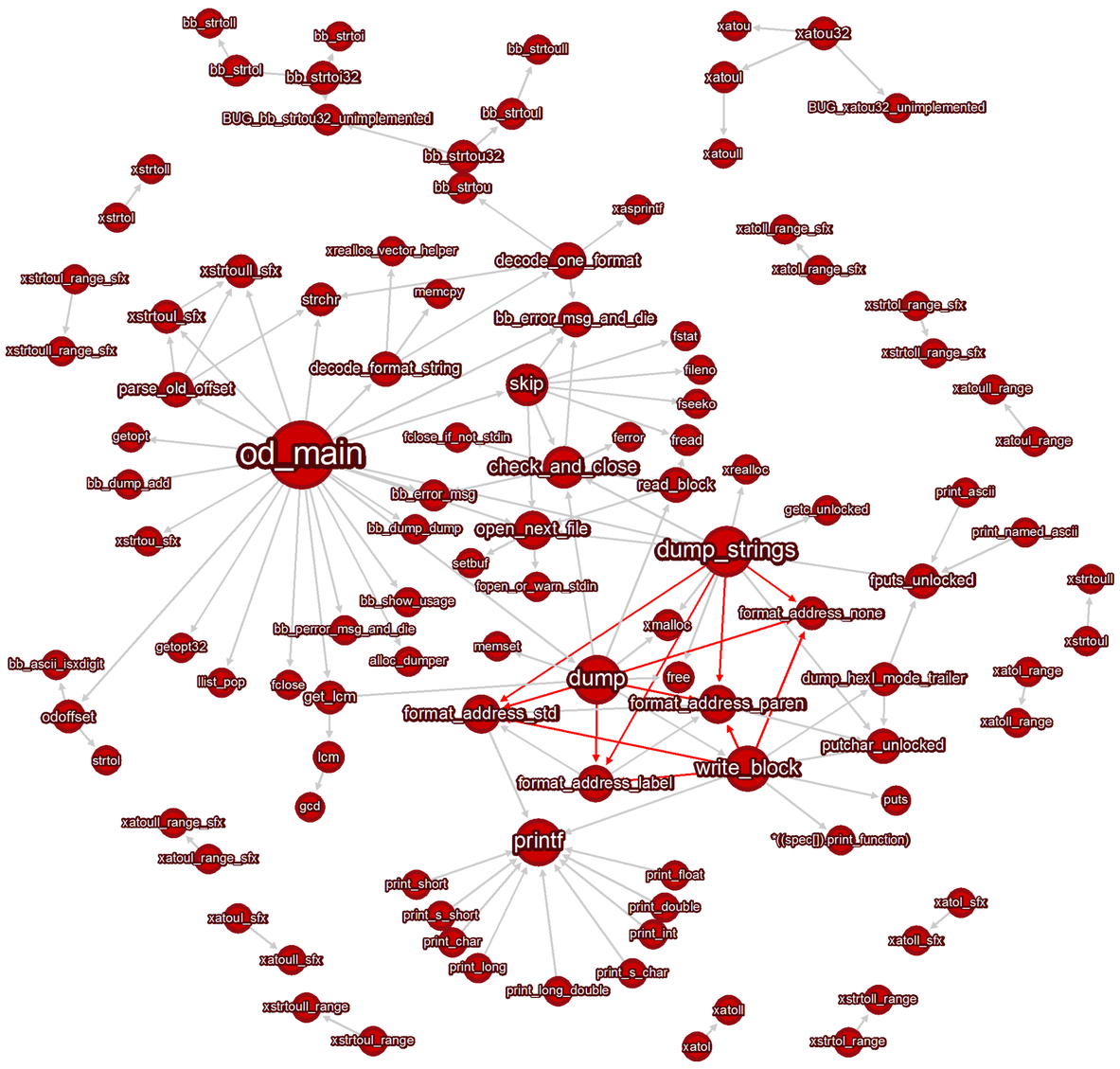
Our approach

- Track configuration options in C code with TypeChef infrastructure
- Build call graphs from source code of target systems
- Enrich call graphs with configuration knowledge
- Increase accuracy of call graphs with a pointer analysis
- Determine presence conditions for all network elements
- Extract network information from the source code
- Apply network science metrics to software call graphs
- Cross-validate network metrics with known vulnerabilities
  Study: Linux & OpenSSL (project), Busybox (here)

Partial results

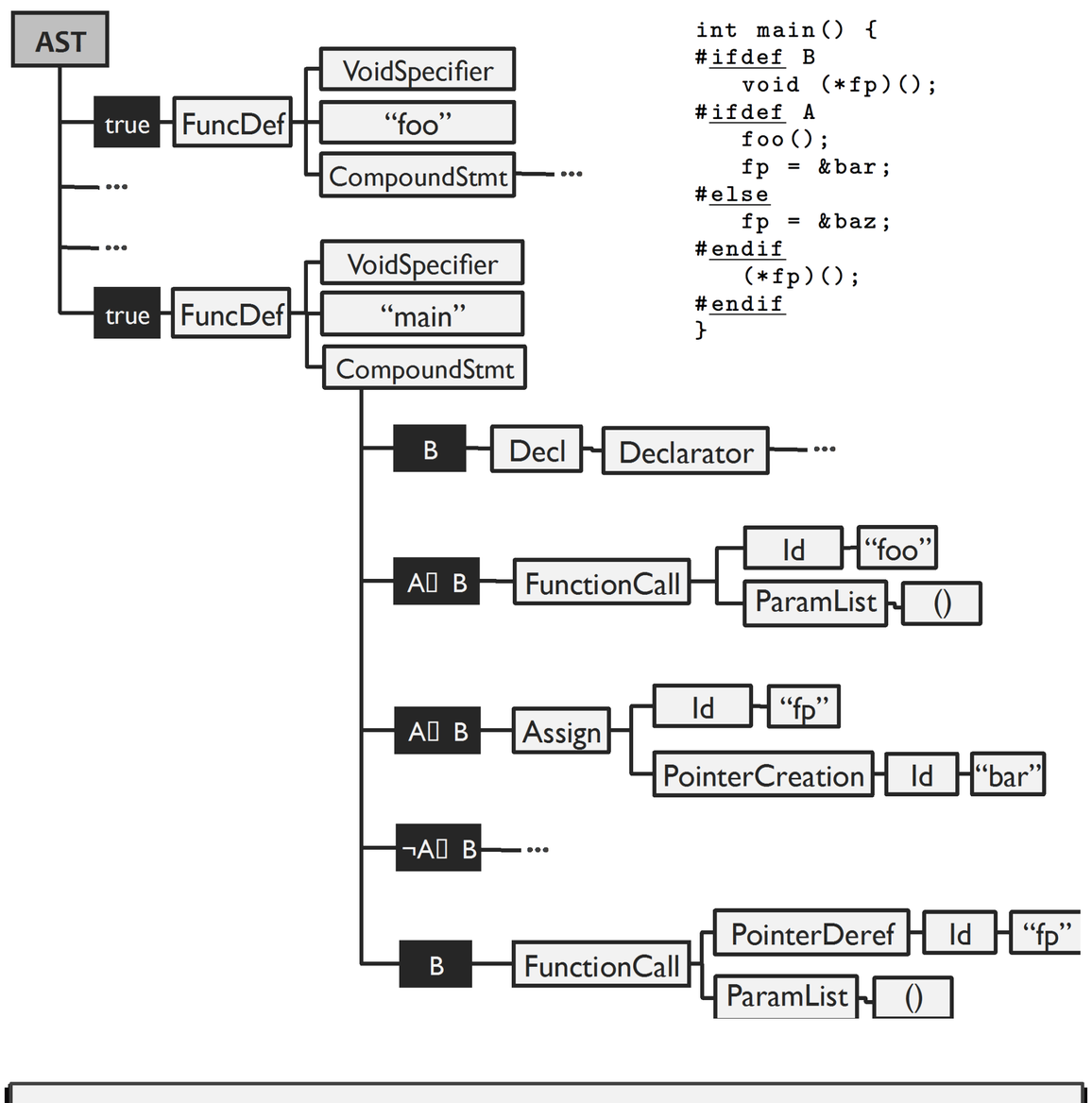
- TypeChef now builds more accurate call graphs (pointer analysis)
- Generated call graphs contain configuration knowledge
- Busybox analysis:
  3318 function definitions with presence conditions (node)
  26719 function calls with presence conditions (edges)
  193 indirect function calls (35% identified by pointer analysis)

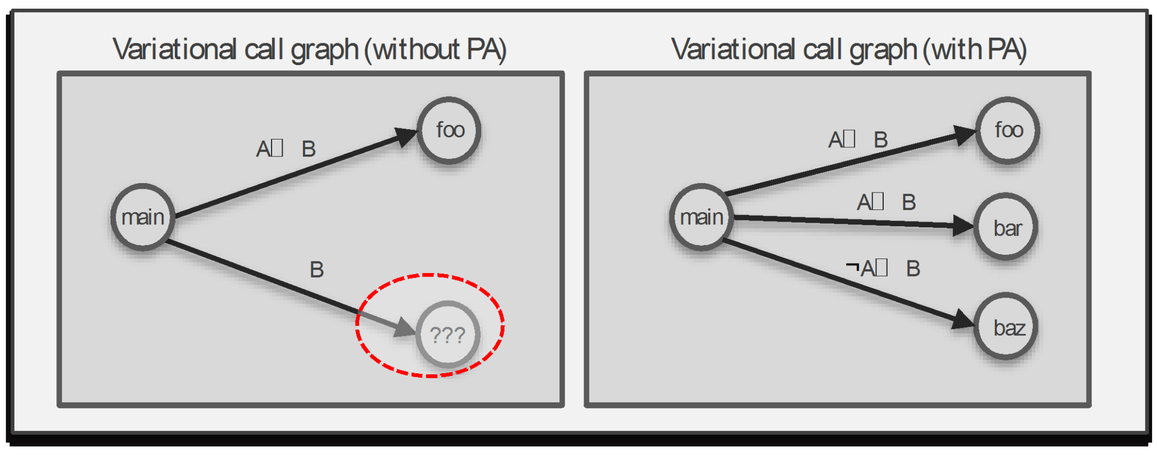
- Call graph example (file: busybox/coreutils/od.c)



Generating variational call graphs from source code

- Build AST representation of source code for all configurations
- Build call graph enriched with pointer analysis





Carnegie Mellon University

UNIVERSITÄT PASSAU

http://hot-sos.org/

The Science of Security initiative is funded by the National Security Agency.