PHP: formal semantics & static analysis

By Daniele Filaretti
4th year PhD @ Imperial College London
Supervised by Sergio Maffeis

FMATS 4 Workshop, Cambridge

Background

- PHP: popular scripting language (server-side)
  - static analysis of PHP: XSS/SQLi, types, bug patterns,...
  - existing analyzers not good enough (poor/ unsound language model)
  - static analysis (of scripting languages) is hard!

The problem

- PHP poorly understood (no formal semantics, confusing docs, no spec, different implementations etc.)
  - existing tools not based on precise meaning of the language
  - their quality depends on their developer's (approximate, informal) understanding of the language

Our goal

- Really understand the language (formally)
- use this knowledge as a foundation to build better (and trusted) tools (static analyzers, model checkers, compilers etc.)

An executable formal semantics of PHP

- A mathematical model of the language
- defined in the term-rewriting based semantics framework “K”:
- formal and executable (thus, testable)
- Should we trust? Yes: tested against reference PHP test suite (Zend)

General purpose static analysis for PHP

- abstract interpretation: derive correct static analyzers from language semantics
- updated semantics, domain as a parameter (execution, type check, taint check, etc.)
- extensible: user can add new domains
- execution and analysis: unified framework

Domains

- Concrete (normal execution)
- Signs (toy domain, signs of variables)
- Types (by Raphaël Rieu-Helft)
- Next: taint analysis, ...

References

- "An Executable Formal Semantics for PHP" by D. Filaretti & S. Maffeis, ECOOP 2014
- http://www.phpsemantics.org (sources + web interface)
- The K Framework: http://www.kframework.org/index.php/Main_Page

Thank you.