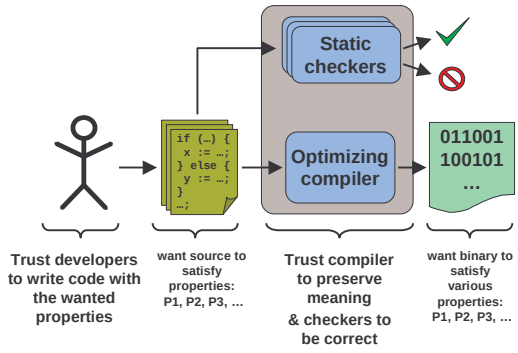
Trustworthy compilation

Sorin Lerner
University of Washington
(work with Todd Millstein, Erika Rice and Craig Chambers)

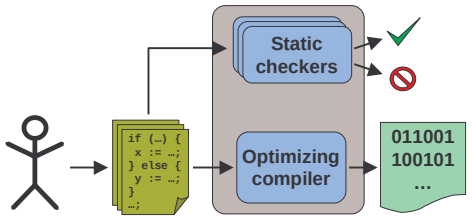
Trustworthy software

- Various guarantees that we may want a software system to have:
  - reliability (does not crash)
  - safety (cannot be hijacked to do “bad” things)
  - secure (does not divulge private information)
  - obeys certain policies
- How can we guarantee that a software system has these properties?

Trustworthy software

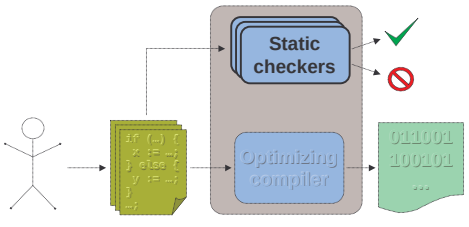


Trustworthy software



Our approach:
- Write analyses and transformations in a domain specific language called Rhodium
- Check them for correctness automatically

Trustworthy software



Our approach:
- Write analyses and transformations in a domain specific language called Rhodium
- Check them for correctness automatically

Static analysis for information flow

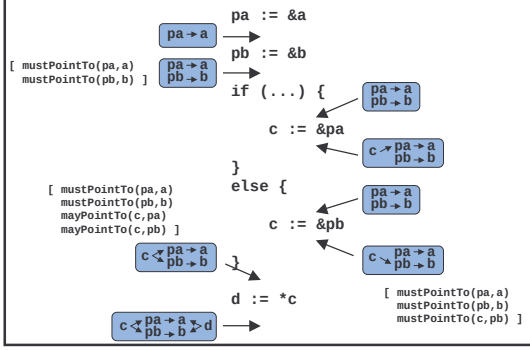
- “High-security input never affects low-security output”
- Enforceable via sound (incomplete) dataflow analysis
  - L ≤ H, assign each variable a level `sec(x)`

`e1 + e2`	`x := e`	`if(x) e;`
`max(sec(e1), sec(e2))`	`sec(e) if sec(e)≤sec(x)`	`sec(x) if sec(x)≤sec(e)`

`*x := e`
`sec(e) if ???`

- requires pointer information
- pointer analyses are hard to get right

Pointer analysis



Pointer analysis in Rhodium

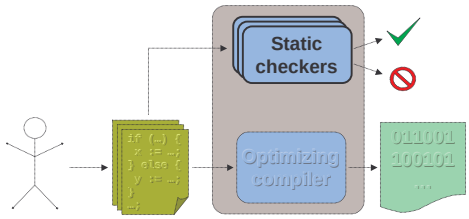
define fact *mustPointTo(X, Y)*
with meaning " $X == \&Y$ **"**

if *currStmt* $==[X := \&Y]$
then *mustPointTo(X,Y)* **after** *currStmt*

if (*mustPointTo(X,Y)* **before** *currStmt*) **and**
(*doesNotModify(X)* **at** *currStmt*)
then *mustPointTo(X,Y)* **after** *currStmt*

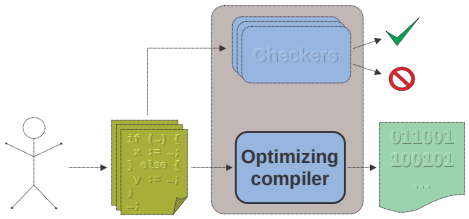
- Leverage domain specific language to allow various kinds of automated processing:
  - run Rhodium analyses
  - automatically check Rhodium analyses for correctness using an automated theorem prover

Trustworthy software



Our approach:
- Write analyses and transformations in a domain specific language called Rhodium
- Check them for correctness automatically

Trustworthy software



Our approach:
- Write analyses and transformations in a domain specific language called Rhodium
- Check them for correctness automatically

Optimizations in Rhodium

if (*currStmt* $==[*X := Z]$) **and**
(*mustPointTo(X,Y)* **before** *currStmt*
)
then transform *currStmt* **to** $[Y := Z]$

- We can check Rhodium optimizations for correctness automatically using an automated theorem prover
- Separate profitability from correctness

Current status and future work

- Current status of Rhodium
  - a language for writing analyses and optimizations over a realistic C-like language
  - automated correctness checker
  - implemented and checked a variety of analyses and optimizations in Rhodium
- Future work
  - add support to Rhodium for writing checkers
  - increase expressiveness
  - efficient execution engine
  - infer rules automatically