Towards the Synthesis of Provably Secure Programs

(work-in-progress)

Xiaomo Yao

xiaomoy@kth.se

Department of Theoretical Computer Science (TCS)
KTH Royal Institute of Technology, Sweden

6 Aug 2019

Program Correctness

What is **correctness**?

Let $\mathcal{P}$, $\mathcal{Q}$ be logical formulas, c be an imperative program, the Hoare triple $\{\mathcal{P}\}c\{\mathcal{Q}\}$ holds.

Problem of deductive **verification**:

Given $\mathcal{P}$, $\mathcal{Q}$, and c, show that $\vdash \{\mathcal{P}\}c\{\mathcal{Q}\}$.

Problem of deductive **synthesis**:

Given $\mathcal{P}$ and $\mathcal{Q}$, find a program c such that $\models \{\mathcal{P}\}c\{\mathcal{Q}\}$.
(and its coherent proof $\vdash \{\mathcal{P}\}c\{\mathcal{Q}\}$)

Software Security

Security obligation:

- The program must also satisfy a security invariant $\mathcal{I}_s$.
  (e.g., information confidentiality, memory isolation, . . .)
- Find a program c such that $\models \{\mathcal{P} \wedge \mathcal{I}_s\} c \{\mathcal{Q} \wedge \mathcal{I}_s\}$.

$$\textit{Synthesizer}: \underbrace{\langle\langle \mathcal{P}, \mathcal{Q}\rangle, \mathcal{I}_s\rangle}_{\substack{\textit{functionality}\\ \textit{spec}} \times \substack{\textit{security}\\ \textit{spec}}} \mapsto \underbrace{\langle c, p\rangle}_{\textit{program} \times \textit{proof}}$$

Challenge: "Synthesis in the Large" is hard.

Compositionality

COMPOSITIONALITY. If two programs preserve the same security invariant $\mathcal{I}_s$, then the sequential composition of them preserves $\mathcal{I}_s$.

i.e.,

$$\frac{\{\mathcal{P} \wedge \mathcal{I}_s\} c_1 \{\mathcal{Q} \wedge \mathcal{I}_s\} \quad \{\mathcal{Q} \wedge \mathcal{I}_s\} c_2 \{\mathcal{R} \wedge \mathcal{I}_s\}}{\{\mathcal{P} \wedge \mathcal{I}_s\} c_1; c_2 \{\mathcal{R} \wedge \mathcal{I}_s\}.}$$

Idea:

- Decompose specs and programs:
  $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_n, \mathcal{Q}$ and $c = c_0; c_1; \ldots; c_n$
- "Synthesis in the Small" is feasible!

Approach

Deductive program synthesis

Prior work (state-of-the-art):

- Proof-theoretic synthesis by invariant inference (Srivastava et al. POPL'10)
- *Synthetic* Separation Logic (Polikarpova and Sergey POPL'19)

Roadmap

What's in the future?

A program synthesis framework with *security* obligations

- for a simple heap-manipulating imperative language
- program and proof co-generation

Potential synthesizing targets:

- trustworthy embedded system components
- simple separation kernel targeting RISC-V
- (many more...)

Thanks.