Relating Cryptography and Polymorphism

Eijiro Sumii

Joint Work with Benjamin Pierce

University of Pennsylvania

Main Result

- Adaptation of *relational parametricity* from polymorphic λ-calculus to *cryptographic l-calculus*

$$\vdash e \sim e' : \tau \;\Rightarrow\; e \approx e'$$

  – useful for reasoning about programs using encryption

    • E.g., (in)correctness proof of security protocols

Outline

- Background
- Parametricity for Type Abstraction
- Cryptographic λ-Calculus
- Parametricity for Encryption
- Current Status and Future Work

Two Approaches to Information Hiding

- Type abstraction
  - conceals the types of data
    - existential types, universal types, modules, packages, etc.
- Encryption
  - obfuscates the values of data

Example

- Using type abstraction:

$$\text{pack int}, \langle 3, \lambda x.\ x \bmod 2\rangle \ \text{as}\ \exists\alpha.\alpha\times(\alpha\to\text{int})$$

- Using encryption:

$$\text{new k in } \langle \{3\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle$$

new k in ...

generate a fresh key k

$\{v\}_k$

a value v encrypted by the key k

Secrecy as Non-Interference as Contextual Equivalence

- *Relational parametricity* [Reynolds 83] (or *representation independence* [Mitchell 86]):

$$\begin{aligned} &\text{pack int}, \langle 3, \lambda x.\ x \bmod 2\rangle \\ &\qquad \text{as } \exists\alpha.\alpha\times(\alpha\to\text{int}) \\ \approx\ &\text{pack int}, \langle 1, \lambda x.\ x \bmod 2\rangle \\ &\qquad \text{as } \exists\alpha.\alpha\times(\alpha\to\text{int}) \end{aligned}$$

- This work:

$$\begin{aligned} &\text{new } k \text{ in } \langle \{3\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle \\ \approx\ &\text{new } k \text{ in } \langle \{1\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle \end{aligned}$$

Outline

- Background
- Parametricity for Type Abstraction
- Cryptographic λ-Calculus
- Parametricity for Encryption
- Current Status and Future Work

Principle of Parametricity: "Related" Values are Equivalen

$$\varphi \vdash e \sim e' : \tau \quad \text{for some } \varphi$$

$$\Downarrow$$

? f e? =? f e'? for any $f : \tau \rightarrow \text{bool}$

- $\sim$ is defined by induction on τ
- φ defines the case of abstract types, mapping each free type variable to a relation between values of its concrete types

Definition of the Logical Relation (1/2)

- Values of a base type (e.g. int) are related iff they are equal
- Functions are related iff they map related arguments to related results
- Pairs are related iff their elements are respectively related

Definition of the Logical Relation (2/2)

- Packages are related iff their implementations can be related

  $\varphi \vdash \text{pack } \tau, v \text{ as } \exists\alpha.\sigma \sim$
  $\quad \text{pack } \tau', v' \text{ as } \exists\alpha.\sigma : \exists\alpha.\sigma \iff$

  $\varphi, \alpha \mapsto r \vdash v[\tau/\alpha] \sim v'[\tau'/\alpha] : \sigma$
  $\quad \text{for some } r \subseteq \tau \times \tau'$

- Values of an abstract type α are related iff they are related by $\varphi(\alpha)$

  $\varphi \vdash v \sim v' : \alpha \iff (v, v') \in \varphi(\alpha)$

Example

$\alpha \mapsto \{(3,1)\} \vdash$

$\langle 3, \lambda x.\ x \bmod 2\rangle \sim \langle 1, \lambda x.\ x \bmod 2\rangle$

$: \alpha \times (\alpha \to \text{int})$

$$\Downarrow$$

$\vdash$ pack int, $\langle 3, \lambda x.\ x \bmod 2\rangle$

as $\exists\alpha.\alpha\times(\alpha\to\text{int})$

$\sim$ pack int, $\langle 1, \lambda x.\ x \bmod 2\rangle$

as $\exists\alpha.\alpha\times(\alpha\to\text{int})$: $\exists\alpha.\alpha\times(\alpha\to\text{int})$

Outline

- Background
- Parametricity for Type Abstraction
- Cryptographic λ-Calculus
- Parametricity for Encryption
- Current Status and Future Work

Cryptographic λ-Calculus: Syntax and Semantics

Simply-typed call-by-value λ-calculus + shared-key cryptographic primitives

- e ::= ... | k | $\{e1\}_{e2}$ | let $\{x\}_{e1}$ = e2 in e3 else e4
- τ ::= ... | key | bits(τ)

let $\{x\}_k = \{v\}_{k'}$ in e else e'
$\rightarrow$ e[v/x] if k = k', e' otherwise

Cryptographic λ-Calculus: Typing Rules

$$\Gamma \vdash k : \text{key}$$

$$\frac{\Gamma \vdash e1 : \tau \qquad \Gamma \vdash e2 : \text{key}}{\Gamma \vdash \{e1\}_{e2} : \text{bits}(\tau)}$$

$$\frac{\begin{array}{c}\Gamma \vdash e1 : \text{key} \qquad \Gamma \vdash e2 : \text{bits}(\tau') \\ \Gamma, x : \tau' \vdash e3 : \tau \qquad \Gamma \vdash e4 : \tau\end{array}}{\Gamma \vdash \text{let } \{x\}_{e1} = e2 \text{ in } e3 \text{ else } e4 : \tau}$$

Outline

- Background
- Parametricity for Type Abstraction
- Cryptographic λ-Calculus
- Parametricity for Encryption
- Current Status and Future Work

Parametricity Adapted

$$\varphi \vdash e \sim e' : \tau \quad \text{for some } \varphi$$

$$\Downarrow$$

$$?\ \ f\ e?\ \ =?\ \ f\ e'?\ \ \text{for any } f : \tau \rightarrow \text{bool}$$

$$\text{s.t. } \mathrm{dom}(\varphi) \cap \mathrm{keys}(f) = \emptyset$$

- $\sim$ is defined by induction on τ, meaning that e and e' are equivalent and <u>don't leak the secret keys</u>
- φ defines the case of bits(τ), mapping each secret key to a relation between values encrypted by the key

Definition of the Logical Relation

- Keys are related iff they are equal *and non-secret*

  $\varphi \vdash k \sim k : \text{key} \iff k \notin \text{dom}(\varphi)$

- Values encrypted by a secret key k are related iff they are related by φ(k)

  $\varphi \vdash \{v\}_k \sim \{v'\}_k : \text{bits}(\tau) \iff$

  $(v,v') \in \varphi(k)$ if $k \in \text{dom}(\varphi)$

  $\varphi \vdash v \sim v' : \tau$ if $k \notin \text{dom}(\varphi)$

Example

$k \mapsto \{(3,1)\} \vdash \{3\}_k \sim \{1\}_k : \text{bits(int)}$

$k \mapsto \{(3,1)\} \vdash \lambda\{x\}_k.\ x \bmod 2$

$\quad \sim \lambda\{x\}_k.\ x \bmod 2 : \text{bits(int)} \to \text{int}$

$$\Downarrow$$

$k \mapsto \{(3,1)\} \vdash \langle\{3\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle \sim$

$\quad \langle\{1\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle$

$\quad : \text{bits(int)} \times (\text{bits(int)} \to \text{int})$

N.B.

$\lambda\{x\}_k.\ e \equiv \lambda z.\ \text{let } \{x\}_k = z \text{ in } e \text{ else } \bot$

Outline

- Background
- Parametricity for Type Abstraction
- Cryptographic λ-Calculus
- Parametricity for Encryption
- Current Status and Future Work

Current Status

- Treatment of fresh key generation, adapting [Stark 97]
- (In)correctness proof of a few security protocols, using the following encoding
  - principal = function from messages to messages (with its own continuation)
  - configuration = record of principals and non-secret keys
  - network and scheduler = "right" context
  - attacker = arbitrary context

Fresh Key Generation (1/2)

- Syntax

$$e \ ::= \ \ldots \ | \ \text{new } x \text{ in } e$$

- Semantics

$$e \Downarrow (S)v$$

read as: "the expression e evaluates to the value v, generating the set S of fresh keys"
note that (S) is a binder

Fresh Key Generation (2/2)

- Logical relation

  $\varphi \vdash e \sim e' : \tau \iff$

  $e \Downarrow (\{k_1,\dots,k_n\} \oplus S)\ v_1$,

  $e' \Downarrow (\{k_1,\dots,k_n\} \oplus S')\ v_2$, and

  $\varphi, k_1 \mapsto r_1, \dots, k_n \mapsto r_n \vdash v_1 \sim v_2 : \tau$

  for some $k_1, \dots, k_n, r_1, \dots, r_n$, S and S'

For example,

$\vdash \text{new } k \text{ in } \langle \{3\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle \sim$

$\text{new } k \text{ in } \langle \{1\}_k, \lambda\{x\}_k.\ x \bmod 2\rangle$

$: \text{bits(int)} \times (\text{bits(int)} \to \text{int})$

Future Work

- Recursive functions/types

  cf. [Birkedal & Harper 97], [Crary & Harper], etc.

- Concurrency and distribution

  cf. spi-calculus [Abadi & Gordon 97], evaluation semantics for CCS [Pitts 96], typed equivalence in polymorphic π-calculus [Pierce & Sangiorgi 97], parametricity in linear polymorphic λ-calculus [Pitts 2000], etc.