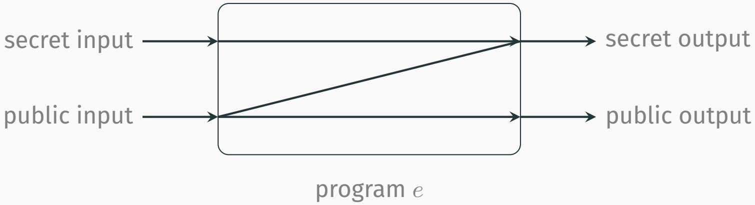
Mechanized Logical Relations for Termination-Insensitive Noninterference

Simon O. Gregersen
joint work with Johan Bay, Amin Timany, and Lars Birkedal

POPL 2021, January 20 - 22, Online

The prevailing basic semantic notion of secure information flow is noninterference.



Program e satisfies termination-insensitive noninterference, abbr. $\mathrm{TINI}(e)$, when

$$e[v_1/x] \Downarrow o_1 \qquad \text{and} \qquad e[v_2/x] \Downarrow o_2 \qquad \text{implies} \qquad o_1 \simeq o_2$$

for all secrets v_1 and v_2.

The problem

Information-flow control enforcement is often specified using a static type system:

$$\Gamma \vdash e : t^{\ell} \qquad \text{implies} \qquad \mathrm{TINI}(e)$$

To be useful, it must support the same features as modern programming languages:

- **higher types**,
- **reference types**,
- **abstract types**,
- ...

The difficulty of proving the type system sound, however, increases.

The problem

Information-flow control enforcement is often specified using a static type system:

$$\Gamma \vdash e : t^{\ell} \qquad \text{implies} \qquad \mathrm{TINI}(e)$$

To be useful, it must support the same features as modern programming languages:

- **higher types**,
- **reference types**,
- **abstract types**,
- ...

The difficulty of proving the type system sound, however, increases.

This work

The main goal of this work is to

- show that such a rich type system satisfies termination-insensitive noninterference
- using a semantic model

  $\Rightarrow$ compositional integration of syntactically well-typed and ill-typed components:

$$\Gamma, x : \tau_2 \vdash e_1 : \tau_1 \qquad \text{and} \qquad e_2 \in [\![\tau_2]\!] \qquad \text{then} \qquad \mathrm{TINI}(e_1[e_2/x])$$

- with full mechanization of all results in Coq

This work

The main goal of this work is to

- show that such a rich type system satisfies termination-insensitive noninterference
- using a semantic model

  $\Rightarrow$ compositional integration of syntactically well-typed and ill-typed components:

  $$\Gamma, x : \tau_2 \vdash e_1 : \tau_1 \qquad \text{and} \qquad e_2 \in [\![\tau_2]\!] \qquad \text{then} \qquad \mathrm{TINI}(e_1[e_2/x])$$

- with full mechanization of all results in Coq

Example (Multiplying by zero)

$$\lambda v.\, v * 0$$

cannot be syntactically typed at $\mathbb{N}^\top \to \mathbb{N}^\perp$.

Example (Temporary explicit leak)

$$\textsf{let}\ x = !\, l\ \textsf{in}\ l \leftarrow !\, h;\ \ldots;\ l \leftarrow x$$

is not syntactically well-typed.

More interesting examples found at the end of the presentation and in the paper.

Example (Multiplying by zero)

$$\lambda v.\, v * 0$$

cannot be syntactically typed at $\mathbb{N}^\top \to \mathbb{N}^\perp$.

Example (Temporary explicit leak)

$$\mathsf{let}\; x = !\, l \;\mathsf{in}\; l \leftarrow !\, h;\ \ldots;\ l \leftarrow x$$

is not syntactically well-typed.

More interesting examples found at the end of the presentation and in the paper.

Language

$$
\begin{aligned}
\tau &::= t^{\ell} \\
t &::= \mathbb{B} \mid \mathbb{N} \mid \tau \times \tau \mid \tau + \tau \mid \\
&\qquad \tau \xrightarrow{\ell} \tau \mid \mathsf{ref}(\tau) \mid \alpha \mid \forall_{\ell}\, \alpha.\, \tau \mid \forall\!\!\!/_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\, \alpha.\, \tau \\
\ell &::= \kappa \mid l \in \mathcal{L} \mid \ell \sqcup \ell
\end{aligned}
$$

Language

$$\begin{aligned}
\tau &::= t^{\ell} \\
t &::= \mathbb{B} \mid \mathbb{N} \mid \tau \times \tau \mid \tau + \tau \mid \\
&\qquad \tau \xrightarrow{\ell} \tau \mid \mathsf{ref}(\tau) \mid \alpha \mid \forall_{\ell}\, \alpha.\, \tau \mid \forall\!\!\!/_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\, \alpha.\, \tau \\
\ell &::= \kappa \mid l \in \mathcal{L} \mid \ell \sqcup \ell
\end{aligned}$$

Consider if $secret$ then $f\ ()$ — if f has public side-effects we would leak $secret$.

Language

$$
\begin{aligned}
\tau &::= t^{\ell} \\
t &::= \mathbb{B} \mid \mathbb{N} \mid \tau \times \tau \mid \tau + \tau \mid \\
&\qquad \tau \xrightarrow{\ell} \tau \mid \mathsf{ref}(\tau) \mid \alpha \mid \forall_{\ell}\, \alpha.\, \tau \mid \forall\!\!\!/_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\, \alpha.\, \tau \\
\ell &::= \kappa \mid l \in \mathcal{L} \mid \ell \sqcup \ell
\end{aligned}
$$

Consider if $secret$ then $f\ ()$ — if f has public side-effects we would leak $secret$.

For this presentation, we consider $\mathcal{L} = \{\bot, \top\}$ where $\bot \sqsubseteq \top$ and $\top \not\sqsubseteq \bot$.

Typing judgment

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau$$

Typing judgment

Term-level context

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau$$

Typing judgment

Term-level context

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau$$

Type-level context

Typing judgment

Term-level context

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau$$

Type-level context

Label context

Typing judgment

Term-level context

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau$$

Type-level context

Program counter label

Label context

Type system

T-IF

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1, 2\} \,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\ e\ \textsf{then}\ e_1\ \textsf{else}\ e_2 : \tau}$$

T-STORE

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\bot}}$$

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : (\forall_{\ell_e} \alpha . \tau)^{\bot}}$$

Type system

T-IF

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1, 2\} \,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\; e \;\textsf{then}\; e_1 \;\textsf{else}\; e_2 : \tau}$$

T-STORE

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\perp}}$$

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : (\forall_{\ell_e} \alpha.\, \tau)^{\perp}}$$

Type system

T-IF

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1,2\} \,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\ e\ \textsf{then}\ e_1\ \textsf{else}\ e_2 : \tau}$$

T-STORE

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\perp}}$$

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e}\, \alpha .\, \tau\right)^{\perp}}$$

Theorem (Termination-Insensitive Noninterference)

If

$$x : \mathbb{B}^\top \vdash_\bot e : \mathbb{B}^\bot, \qquad \vdash_\bot v_1 : \mathbb{B}^\top, \qquad \textit{and} \qquad \vdash_\bot v_2 : \mathbb{B}^\top$$

then

$$(\emptyset, e[v_1/x]) \to^* (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \to^* (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

If

$$x : \mathbb{B}^{\top} \vdash_{\bot} e : \mathbb{B}^{\bot}, \qquad \vdash_{\bot} v_1 : \mathbb{B}^{\top}, \qquad \textit{and} \qquad \vdash_{\bot} v_2 : \mathbb{B}^{\top}$$

then

$$(\emptyset, e[v_1/x]) \rightarrow^* (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \rightarrow^* (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

If

$$x : \mathbb{B}^\top \vdash_\bot e : \mathbb{B}^\bot, \qquad \vdash_\bot v_1 : \mathbb{B}^\top, \qquad \textit{and} \qquad \vdash_\bot v_2 : \mathbb{B}^\top$$

then

$$(\emptyset, e[v_1/x]) \rightarrow^* (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \rightarrow^* (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

If

$$x : \mathbb{B}^\top \vdash_\perp e : \mathbb{B}^\perp, \qquad \vdash_\perp v_1 : \mathbb{B}^\top, \qquad \textit{and} \qquad \vdash_\perp v_2 : \mathbb{B}^\top$$

then

$$(\emptyset, e[v_1/x]) \to^* (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \to^* (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Our approach

We set up a binary (logical) relation

$$\Xi \mid \Psi \mid \Gamma \vDash e_1 \approx e_2 : \tau$$

such that

$$\begin{aligned} \Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \quad &\Rightarrow \quad \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \\ \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \quad &\Rightarrow \quad \mathrm{TINI}(e) \end{aligned}$$

However, this requires manipulating and defining a complex semantic model.

Our approach cont'd

We combat this complexity by using the separation logic framework Iris.

- Convenient modalities to express the relation,
- High-level logic to reason within, and
- Coq formalization and the Iris Proof Mode to mechanize proofs.

Our approach cont'd cont'd

Existing works on "logical" logical relations prove (contextual) refinements.

Intuitively, e_1 refines e_2 if

$$e_1 \to^* v_1 \quad \Rightarrow \quad e_2 \to^* v_2 \quad \land \quad v_1 \approx v_2.$$

Our approach cont'd cont'd

Existing works on "logical" logical relations prove (contextual) refinements.

Intuitively, e_1 refines e_2 if

$$e_1 \rightarrow^* v_1 \quad \Rightarrow \quad e_2 \rightarrow^* v_2 \quad \wedge \quad v_1 \approx v_2.$$

However, we need a termination-insensitive notion:

$$e_1 \rightarrow^* v_1 \quad \wedge \quad e_2 \rightarrow^* v_2 \quad \Rightarrow \quad v_1 \approx v_2.$$

Our approach cont'd cont'd

Existing works on "logical" logical relations prove (contextual) refinements.

Intuitively, e_1 refines e_2 if

$$e_1 \to^* v_1 \quad \Rightarrow \quad e_2 \to^* v_2 \quad \wedge \quad v_1 \approx v_2.$$

However, we need a termination-insensitive notion:

$$e_1 \to^* v_1 \quad \wedge \quad e_2 \to^* v_2 \quad \Rightarrow \quad v_1 \approx v_2.$$

For this, we define a novel theory of modal weakest preconditions.

Semantic model

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are publicly equivalent and as a

Unary relation for characterizing terms that do not have public side-effects.

Semantic model

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are publicly equivalent and as a

Unary relation for characterizing terms that do not have public side-effects.

Consider

$$\vDash \quad \textsf{if}\ v\ \textsf{then}\ e_1\ \textsf{else}\ e_2 \quad \approx \quad \textsf{if}\ v'\ \textsf{then}\ e_1\ \textsf{else}\ e_2 \quad : \quad t^{\top}$$

where $\vDash v \approx v' : \mathbb{B}^{\top}$ meaning $v, v' \in \{\textsf{true}, \textsf{false}\}$.

Semantic model

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are publicly equivalent and as a

Unary relation for characterizing terms that do not have public side-effects.

Consider

$$\vDash \quad \text{if } v \text{ then } e_1 \text{ else } e_2 \quad \approx \quad \text{if } v' \text{ then } e_1 \text{ else } e_2 \quad : \quad t^\top$$

where $\vDash v \approx v' : \mathbb{B}^\top$ meaning $v, v' \in \{\text{true}, \text{false}\}$. This means proving, e.g.,

$$\vDash \quad e_1 \quad \approx \quad e_2 \quad : \quad t^\top$$

Semantic model

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are publicly equivalent and as a

Unary relation for characterizing terms that do not have public side-effects.

Consider

$$\vDash \quad \textsf{if}\ v\ \textsf{then}\ e_1\ \textsf{else}\ e_2 \quad \approx \quad \textsf{if}\ v'\ \textsf{then}\ e_1\ \textsf{else}\ e_2 \quad : \quad t^{\top}$$

where $\vDash v \approx v' : \mathbb{B}^{\top}$ meaning $v, v' \in \{\textsf{true}, \textsf{false}\}$. This means proving, e.g.,

$$\vDash \quad e_1 \quad \approx \quad e_2 \quad : \quad t^{\top}$$

Crucially, they may not modify public references.

Semantic typing

Recall

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \qquad \Rightarrow \qquad \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau$$

$$\Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \qquad \Rightarrow \qquad \mathrm{TINI}(e)$$

Importantly, the semantic relation is not defined in terms of the syntactic relation.

Semantic typing

Recall

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \qquad \Rightarrow \qquad \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau$$
$$\Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \qquad \Rightarrow \qquad \mathrm{TINI}(e)$$

Importantly, the semantic relation is not defined in terms of the syntactic relation.

At the same time,

$$x : \tau_2 \vDash e_1 \approx e_1 : \tau_1 \quad \text{and} \quad \vDash e_2 \approx e_2 : \tau_2$$

implies

$$\vDash e_1[e_2/x] \approx e_1[e_2/x] : \tau_1$$

Value-dependent classification

Consider

$$\begin{aligned} \textit{valDep} \triangleq \lambda f.\, & \mathsf{let}\, d = \mathsf{ref}(\mathsf{true}, secret)\ \mathsf{in} \\ & f\ d; \\ & \mathsf{let}\ (b, v) = !\, d\, \mathsf{in} \\ & \mathsf{if}\ b\ \mathsf{then}\ 42\ \mathsf{else}\ v \end{aligned}$$

Value-dependent classification

Consider

$$\begin{aligned} \textit{valDep} \triangleq \lambda f.\, & \mathsf{let}\, d = \mathsf{ref}(\mathsf{true}, secret)\, \mathsf{in} \\ & f\ d; \\ & \mathsf{let}\, (b, v) = !\, d\, \mathsf{in} \\ & \mathsf{if}\, b\, \mathsf{then}\, 42\, \mathsf{else}\, v \end{aligned}$$

The program does not syntactically type check at $\mathbb{N}^{\perp}$

Value-dependent classification

Consider

$$\begin{aligned} \mathit{valDep} \triangleq \lambda f.\, & \mathsf{let}\, d = \mathsf{ref}(\mathsf{true}, \mathit{secret})\, \mathsf{in} \\ & f\ d; \\ & \mathsf{let}\, (b, v) = !\, d\, \mathsf{in} \\ & \mathsf{if}\, b\, \mathsf{then}\, 42\, \mathsf{else}\, v \end{aligned}$$

The program does not syntactically type check at $\mathbb{N}^{\perp}$ but, ideally,

$$\mathit{secret} : \mathbb{N}^{\top} \vDash \mathit{valDep}\ f \approx \mathit{valDep}\ f : \mathbb{N}^{\perp}$$

for “well-behaved” f. We can use the logic to express and prove these requirements.

Value-dependent classification cont'd

However, this burdens the client with proof obligations. Instead, we can exploit existential types to conceal the proof obligations. E.g.,

$$\begin{aligned}\textit{valDepPack} \triangleq\ & \mathsf{let}\ get = \lambda\, d.\ \mathsf{let}\ (b, v) = !\, d\ \mathsf{in}\ \mathsf{if}\ b\ \mathsf{then}\ \mathsf{inj}_1\ v\ \mathsf{else}\ \mathsf{inj}_2\ v\ \mathsf{in} \\ & \mathsf{let}\ setL = \lambda\, d, v.\ d \leftarrow (\mathsf{false}, v)\ \mathsf{in} \\ & \mathsf{let}\ setH = \lambda\, d, v.\ d \leftarrow (\mathsf{true}, v)\ \mathsf{in} \\ & \mathsf{pack}\ (\mathsf{ref}(\mathsf{true}, secret), get, setL, setH)\end{aligned}$$

for which it holds

$$\begin{aligned} secret : \mathbb{N}^{\top} \vDash \textit{valDepPack} \approx \textit{valDepPack} :\ & \\ \exists \alpha.\ \Big(\alpha^{\bot} \times \Big(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\top} + \mathbb{N}^{\bot}\Big) \times \Big(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\bot} \xrightarrow{\bot} 1\Big) \times \Big(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\top} \xrightarrow{\bot} 1\Big)\Big) & \end{aligned}$$

Conclusion

In summary, we have

- defined a novel semantic model of an expressive IFC type system with support for impredicative polymorphism, label polymorphism, recursive types, and general references,
    - unary and binary logical-relations models
    - a theory of Modal Weakest Preconditions
- showed that the type system entails termination-insensitive noninterference, and
- illustrated how the model can be used to reason about syntactically ill-typed but semantically secure code with compositional integration.

Conclusion

In summary, we have

- defined a novel semantic model of an expressive IFC type system with support for impredicative polymorphism, label polymorphism, recursive types, and general references,
    - unary and binary logical-relations models
    - a theory of Modal Weakest Preconditions
- showed that the type system entails termination-insensitive noninterference, and
- illustrated how the model can be used to reason about syntactically ill-typed but semantically secure code with compositional integration.

Thank you for watching

gregersen@cs.au.dk

`https://cs.au.dk/~gregersen/papers/2021-tiniris.pdf`

`https://github.com/logsem/iris-tini`