DERIVING AND PROVING ABSTRACT NON-INTERFERENCE

Roberto Giacobazzi and Isabella Mastroeni

Dipartimento di Informatica
Università di Verona
Italy

Paris, February 20th, 2004
The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed

- Attack = disclosing properties of confidential data
- Access control by declaring data privileges!
The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed

⇒ Access control methods do not put constraints on how the information is propagated!
The Problem

The problem: Protect data confidentiality from erroneous/malicious attacks while data are processed.

Description of the problem:

- Typing of data (and variables) in private (\(H\)) and public (\(L\));
- **Non-Interference**: to prevent the results of the computation from leaking even partial information about private inputs!
  - *Explicit flow*: caused by directly passing private data to a public variable: \(l := 2 \times h\);
  - *Implicit flow*: arise from control structure of the program: 
    ```
    while h do l := l + 1; h := h - 1.
    ```
- We consider only *terminating* computations!
IT IS ESSENTIAL TO KNOW HOW MUCH AN ATTACKER MAY LEARN FROM A PROGRAM!

Goal: Automatically generate certificates about secure information flows
- Design of accurate security polices
- Static program analysis & verification techniques (types, CFA, DFA, ...)

The Goal
The Goal

IT IS ESSENTIAL TO KNOW HOW MUCH AN ATTACKER MAY LEARN FROM A PROGRAM!

Goal: Automatically generate certificates about secure information flows

State of the art: Standard non-interference is far too restrictive
- No sensitive information can be disclosed
- Any change upon confidential data has not to be revealed by public ones
- Rigid security policy: L can flow into H but H cannot flow into L
  [Denning and Denning ’77]
The Goal

It is essential to know how much an attacker may learn from a program!

**Goal:** Automatically generate certificates about secure information flows

**State of the art:** Standard non-interference is far too restrictive

**Question:** Is there a way to characterize what kind of information flows?

- Characterize the secrecy degree of a program
- $H$ can flow into $L$ unless a given property of $H$ is disclosed
- Weakening standard non-interference (a challenge in language-based security [Sabelfeld & Myers ’03])
IDEA: Attackers as Abstract Interpretations

while h do (l := l + 2; h := h - 1)
IDEA: Attackers as Abstract Interpretations

while \( h \) do \( (l := l + 2; \ h := h - 1) \)

- There is an (implicit/absolute) flow from \( h \) into \( l \)
- The parity of \( l \) is not affected by any change of \( h \)
- ... no information flow for \textit{parity}!
IDEA: Attackers as Abstract Interpretations

while h do (l := l + 2; h := h − 1)

The idea: Abstract non-interference

- Attackers as program analyzers
  - Attackers can analyze I/O behaviour of public data
  - Attackers perform “static” program analyses

⇒ Abstract interpretation is a general method for specifying approximate semantics of programs [Cousot & Cousot ’77]

Attackers are abstract interpretations of program semantics
IDEA: Attackers as Abstract Interpretations

The idea: Abstract non-interference

Main results:
- Generalizing non-interference relatively to the attacker’s power
- Making non-interference parametric on the attacker’s point of view
- Checking abstract non-interference by abstract interpretation
- Systematic method for deriving attackers for programs by modifying abstractions
- Abstract Robust Declassification
Related works

Refining security policies by constraining attackers
Characterizing released information
Related works

Refining security policies by constraining attackers

- Complexity:
  - Security levels corresponding to how complex is attacking the program

[Lowe ’02]
Related works

Refining security policies by constraining attackers

- **Complexity:**
  - Security levels corresponding to how complex is attacking the program
    [Lowe ’02]

- **Quantitative measure:**
  - An absolute (approximate) quantitative evaluation of information leakage (number of statistical tests to disclose properties)
    [Di Pierro et al. ’02]
Related works

Characterizing released information

Quantitative measure:
- Quantification of the information flowed by information theory
  [D. Clark et al. ’03]
Related works

Characterizing released information

Quantitative measure:

- Quantification of the information flowed by information theory
  [D. Clark et al. ’03]

Robust declassification:

- The observational capability of the attacker is characterized by equivalence relations, then the information released is identified and declassified.
  [Zdancewic and Myers ’01]
Standard non-interference

“One group of users [...] is noninterfering with another group of users if what the first group does [...] has no effect on what the second group of users can see” [Goguen & Meseguer '82]

Standard non-interference

\[ \forall l : L, \forall h_1, h_2 : H. \mathbb{[[P]]}(h_1, l)^L = \mathbb{[[P]]}(h_2, l)^L \]
Standard non-interference

\[ \forall l : L, \forall h_1, h_2 : H. \llbracket P \rrbracket (h_1, l)^L = \llbracket P \rrbracket (h_2, l)^L \]

**Example:**

\[ \text{while } h \text{ do } (l := l + 2; h := h - 1). \]
Standard non-interference

\[ \forall l : L, \forall h_1, h_2 : H. \llbracket P \rrbracket(h_1, l)^L = \llbracket P \rrbracket(h_2, l)^L \]

**Example:**

\[ \textbf{while } \ h \ \textbf{do} \ (l := l + 2; \ h := h - 1). \]

\[ \begin{align*}
  h = 0, \ l = 1 & \implies l = 1 \\
  h = 1, \ l = 1 & \implies l = 3 \\
  h = n, \ l = 1 & \implies l = 1 + 2n
\end{align*} \]
Standard non-interference

\[ \forall l : L, \forall h_1, h_2 : H. \, [P](h_1, l)^L = [P](h_2, l)^L \]

**Example:**

\[
\text{while } h \text{ do } (l := l + 2; \ h := h - 1).
\]

\[
\begin{align*}
h = 0, \ l = 1 & \leadsto l = 1 \\
h = 1, \ l = 1 & \leadsto l = 3 \\
h = n, \ l = 1 & \leadsto l = 1 + 2n
\end{align*}
\]

If \( l \) is unchanged then \( h \) is 0!

\[ \leadsto \text{There is an information flow from } h \text{ into } l. \]
Standard non-interference

\[ \forall l : \mathbb{L}, \forall h_1, h_2 : \mathbb{H}. \llbracket P \rrbracket (h_1, l)^L = \llbracket P \rrbracket (h_2, l)^L \]

**Example:**

while \( h \) do (\( l := l + 2; \ h := h - 1 \)).

\[
\begin{align*}
 h &= 0, \ l = 1 \quad \Rightarrow \quad l = 1 \\
 h &= 1, \ l = 1 \quad \Rightarrow \quad l = 3 \\
 h &= n, \ l = 1 \quad \Rightarrow \quad l = 1 + 2n
\end{align*}
\]

If \( l \) is unchanged then \( h \) is 0!
\[ \Rightarrow \] There is an information flow from \( h \) into \( l \).

\[ \Rightarrow \] Note that if the input \( l \) is even/odd then the output \( l \) is even/odd!
Abstracting non-interference I

Standard non-interference

$$\forall l : L, \forall h_1, h_2 : H. \; \llbracket P \rrbracket(h_1, l)^L = \llbracket P \rrbracket(h_2, l)^L$$

Consider $$\alpha, \eta \in \text{Abs}(\varphi(L))$$:
Abstracting non-interference

Standard non-interference
\[ \forall l : L, \forall h_1, h_2 : H. \; \llbracket P \rrbracket(h_1, l)^L = \llbracket P \rrbracket(h_2, l)^L \]

Consider \( \alpha, \eta \in \text{Abs}(\phi(V^L)) \):

**Narrow (abstract) non-interference** \([\eta]P(\alpha)\):
\[ \eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1)^L) = \alpha(\llbracket P \rrbracket(h_2, l_2)^L) \]

- No change of \( H \) values and \( \eta \)-equivalent \( L \) values may affect the \( \alpha \) abstraction of \( L \) outputs.

- Possible deceptive interference due to \( \eta \)-undistinguished \( L \) values!

- The more \( \eta \) is precise the less deceptive interference appears
Abstracting non-interference I

Standard non-interference
\[ \forall l : L, \forall h_1, h_2 : H. \; \llbracket P \rrbracket(h_1, l)^L = \llbracket P \rrbracket(h_2, l)^L \]

Consider \( \alpha, \eta \in Abs(\wp(\forall L)) \):

*Narrow (abstract) non-interference* \([\eta]P(\alpha)\):
\[ \eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1)^L) = \alpha(\llbracket P \rrbracket(h_2, l_2)^L) \]

**EXAMPLE:** \([\text{id}]P(\text{Par})\)

\[ P = \text{while } h \text{ do } (l := l + 2; \; h := h - 1). \]

\[ h = 0, \; l = 1 \; \sim \; \text{Par}(l) = \text{odd} \]
\[ h = 1, \; l = 1 \; \sim \; \text{Par}(l) = \text{odd} \]
\[ h = n, \; l = 1 \; \sim \; \text{Par}(l) = \text{odd} \]
Abstracting non-interference I

Standard non-interference

\[ \forall l : L, \forall h_1, h_2 : H. \; \llbracket P \rrbracket(h_1, l) = \llbracket P \rrbracket(h_2, l) \]

Consider \( \alpha, \eta \in \text{Abs}(\wp(V^L)) \):

**Narrow (abstract) non-interference** \([\eta] P(\alpha)\):

\[ \eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1) = \alpha(\llbracket P \rrbracket(h_2, l_2)) \]

**EXAMPLE:** \([i \circ d] P(Par)\)

\[
P = \text{while } h \text{ do } (l := l + 2; \ h := h - 1).
\]

\[
\begin{align*}
\text{h = 0, l = 1 } & \implies \text{Par(l) = odd} \\
\text{h = 1, l = 1 } & \implies \text{Par(l) = odd} \\
\text{h = n, l = 1 } & \implies \text{Par(l) = odd}
\end{align*}
\]

If \( l \) is odd/even then, independently from \( h \), after the execution \( l \) is odd/even!

\( \sim \) There is not an information flow from \( h \) into the parity of \( l \).
Abstracting non-interference I

Standard non-interference
\[ \forall l : L, \forall h_1, h_2 : H. \llbracket P \rrbracket(h_1, l)^L = \llbracket P \rrbracket(h_2, l)^L \]

Consider \( \alpha, \eta \in \text{Abs}(\varphi(\forall^L)) \):

**Narrow (abstract) non-interference** \([\eta]P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1)^L) = \alpha(\llbracket P \rrbracket(h_2, l_2)^L)
\]

**Example II:** \([Par]P(\text{Sign})\)

\[
P = l := 2 \ast l \ast h^2.
\]

\[h = -3, \ l = -2 \ (Par(-2) = \text{even}) \sim \ Sign(l) = -\]
\[h = 1, \ l = -4 \ (Par(-4) = \text{even}) \sim \ Sign(l) = -\]
Abstracting non-interference

Standard non-interference
\[ \forall l \in \mathbb{L}, \forall h_1, h_2 : \mathbb{H}. \ [P](h_1, l)^{L} = [P](h_2, l)^{L} \]

Consider \( \alpha, \eta \in \text{Abs}(\wp(\mathbb{V}^{L})) \):

Narrow (abstract) non-interference \([\eta]P(\alpha)\):
\[ \eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^{L}) = \alpha([P](h_2, l_2)^{L}) \]

Example II: \([Par]P(\text{Sign})\)

\[ P = l := 2 \ast l \ast h^2. \]

\[ h = 1, \ l = 4 \ (Par(4) = \text{even}) \sim \ Sign(l) = + \]
\[ h = 1, \ l = -4 \ (Par(-4) = \text{even}) \sim \ Sign(l) = - \]

The sign of the output \( l \) depends on the sign of the input \( l \)!
\sim \text{There is a DECEPTIVE FLOW!}
Abstracting non-interference I

Standard non-interference
\[ \forall l : L, \forall h_1, h_2 : H. [P](h_1, l)^L = [P](h_2, l)^L \]

Consider \( \alpha, \eta \in Abs(\varphi(\forall^L)) \):

**Narrow (abstract) non-interference** \([\eta]P(\alpha)\):
\[ \eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^L) = \alpha([P](h_2, l_2)^L) \]

**Example II:** \([Par]P(\text{Sign})\)

\[ P = l := 2 * l * h^2. \]

\[ h = 1, \ l = 4 \ (Par(4) = \text{even}) \sim \ Sign(l) = + \]
\[ h = 1, \ l = -4 \ (Par(-4) = \text{even}) \sim \ Sign(l) = - \]

The sign of the output \( l \) depends on the sign of the input \( l \)!
\[ \sim \sim \text{There is a DECEPTIVE FLOW!} \]

\( \Rightarrow \) We compute the semantics on the concrete value of the input \( l \)!
Consider $\alpha, \eta \in \text{Abs}(\varphi(\forall^L))$:

**Narrow (abstract) non-interference** $[\eta]P(\alpha)$:

$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^L) = \alpha([P](h_2, l_2)^L)$
Abstracting non-interference II

Consider \( \alpha, \eta \in \text{Abs}(\varphi(\mathbb{V}^L)) \):

**Narrow (abstract) non-interference** \([\eta]P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1)^L) = \alpha(\llbracket P \rrbracket(h_2, l_2)^L)
\]

**Abstracting non-interference** \( (\eta)P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, \eta(l_1))^L) = \alpha(\llbracket P \rrbracket(h_2, \eta(l_2))^L)
\]

\(\diamond\) No change of \(H\) values may affect the \(\alpha\) abstraction of \(L\) outputs.

\(\diamond\) No deceptive interference due to \(L\) data.
Consider $\alpha, \eta \in \text{Abs}(\wp(\mathbb{N}))$:

**Narrow (abstract) non-interference** $[\eta]P(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1^L)) = \alpha(\llbracket P \rrbracket(h_2, l_2^L))$

**Abstracting non-interference** $(\eta)P(\alpha)$:
$\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, \eta(l_1)))^L = \alpha(\llbracket P \rrbracket(h_2, \eta(l_2))^L)$

**Example:** $(\text{Par})P(\text{Sign})$

\[
P = l := 2 * l * h^2.
\]

$h = -3$, $\text{Par}(l) = \text{even} \Rightarrow \text{Sign}(l) = I \text{ don't know}$

$h = 1$, $\text{Par}(l) = \text{even} \Rightarrow \text{Sign}(l) = I \text{ don't know}$
Consider $\alpha, \eta \in \text{Abs}(\wp(\mathbb{V}^L))$:

**Narrow (abstract) non-interference** $[\eta]P(\alpha)$:

$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^L) = \alpha([P](h_2, l_2)^L)$

**Abstracting non-interference** $\eta P(\alpha)$:

$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, \eta(l_1))^L) = \alpha([P](h_2, \eta(l_2))^L)$

**Example**: $(Par)P(\text{Sign})$

$$P = l := 2 \ast l \ast h^2.$$  

$h = -3$, $Par(l) = \text{even} \not\Rightarrow\Rightarrow Sign(l) = I \text{ don't know}$  

$h = 1$, $Par(l) = \text{even} \not\Rightarrow\Rightarrow Sign(l) = I \text{ don't know}$

$\not\Rightarrow\Rightarrow$ There is not an information flow from $h$ into the sign of $l$. 
Abstracting non-interference II

Consider \( \alpha, \eta \in \text{Abs}(\varphi(\mathcal{V}^L)) \):

**Narrow (abstract) non-interference** \([\eta]P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, l_1)^L) = \alpha(\llbracket P \rrbracket(h_2, l_2)^L)
\]

**Abstracting non-interference** \((\eta)P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha(\llbracket P \rrbracket(h_1, \eta(l_1))^L) = \alpha(\llbracket P \rrbracket(h_2, \eta(l_2))^L)
\]

**Example II**: \((\text{id})P(\text{Par})\)

\[
P = l := l \ast h^2.
\]
\[
h = 2, \ l = 1 \leadsto \text{Par}(l) = \text{even}
\]
\[
h = 3, \ l = 1 \leadsto \text{Par}(l) = \text{odd}
\]
\[
h = n, \ l = 1 \leadsto \text{Par}(l) = \text{Par}(n)
\]
Abstracting non-interference II

Consider $\alpha, \eta \in \text{Abs}(\varphi(\mathbb{W}_L))$:

Narrow (abstract) non-interference $[\eta]P(\alpha)$:
$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^L) = \alpha([P](h_2, l_2)^L)$$

Abstracting non-interference $(\eta)P(\alpha)$:
$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, \eta(l_1))^L) = \alpha([P](h_2, \eta(l_2))^L)$$

Example II: $(\text{id})P(\text{Par})$

$$P = \ l := l \ast h^2.$$  

- $h = 2, \ l = 1 \leadsto \text{Par}(l) = \text{even}$  
- $h = 3, \ l = 1 \leadsto \text{Par}(l) = \text{odd}$  
- $h = n, \ l = 1 \leadsto \text{Par}(l) = \text{Par}(n)$  

$\leadsto$ The parity of $h$ is flowing into $l$!
Abstracting non-interference II

Consider \( \alpha, \eta \in \text{Abs}(\varphi(\varnothing^L)) \):

\textbf{Narrow (abstract) non-interference} \([\eta]P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, l_1)^L) = \alpha([P](h_2, l_2)^L)
\]

\textbf{Abstracting non-interference} \((\eta)P(\alpha)\):
\[
\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, \eta(l_1))^L) = \alpha([P](h_2, \eta(l_2))^L)
\]

\textbf{Example II}: \((\text{id})P(\text{Par})\)

\[
P = l := l \cdot h^2.
\]

\[
\begin{align*}
h = 2, & \quad l = 1 \quad \leadsto \quad \text{Par}(l) = \text{even} \\
h = 3, & \quad l = 1 \quad \leadsto \quad \text{Par}(l) = \text{odd} \\
h = n, & \quad l = 1 \quad \leadsto \quad \text{Par}(l) = \text{Par}(n)
\end{align*}
\]

\(\leadsto\) The parity of \(h\) is flowing into \(l\)!

\(\Rightarrow\) We are looking for flows from any possible property of \(h\) into \(l\)!
Consider $\alpha, \eta \in \text{Abs}(\mathcal{V}^L)$ and $\phi \in \text{Abs}(\mathcal{V}^H)$:

**Abstracting non-interference** $(\eta)P(\alpha)$:

$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P]((h_1, \eta(l_1))^L) = \alpha([P]((h_2, \eta(l_2))^L))$
Abstracting non-interference III

Consider $\alpha, \eta \in Abs(\wp(\mathcal{V}_L))$ and $\phi \in Abs(\wp(\mathcal{V}_H))$:

**Abstracting non-interference** $(\eta)P(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, \eta(l_1))^L) = \alpha([P](h_2, \eta(l_2))^L)$$

**Abstract non-interference** $(\eta)P(\phi \sim\llbracket \alpha \rrbracket)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](\phi(h_1), \eta(l_1))^L) = \alpha([P](\phi(h_2), \eta(l_2))^L)$$

- No change of $\phi$-equivalent $H$ values may affect the $\alpha$ abstraction of $L$ outputs.
- No deceptive interference due to $L$ data;
- $\phi$ does not flow into what $\alpha$ can see on the output.
Abstracting non-interference III

Consider $\alpha, \eta \in Abs(\wp(\mathbb{V}^L))$ and $\phi \in Abs(\wp(\mathbb{V}^H))$:

**Abstracting non-interference** $(\eta)P(\alpha)$:
$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](h_1, \eta(l_1))^L) = \alpha([P](h_2, \eta(l_2))^L)$$

**Abstract non-interference** $(\eta)P(\phi \sim \llbracket \alpha \rrbracket)$:
$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([P](\phi(h_1), \eta(l_1))^L) = \alpha([P](\phi(h_2), \eta(l_2))^L)$$

**Example**: $(id)P(\text{Sign} \sim \llbracket \text{Par} \rrbracket)$

$$P = l := l \cdot h^2.$$ 

$Sign(h) = +$, $l = 1 \sim Par(l) = l \text{ don’t know}$

$Sign(h) = -$ , $l = 1 \sim Par(l) = l \text{ don’t know}$
Consider $\alpha, \eta \in \text{Abs}(\varphi(\mathbb{V}^L))$ and $\phi \in \text{Abs}(\varphi(\mathbb{V}^H))$:

**Abstracting non-interference** $(\eta) \text{P}(\alpha)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\text{P}](h_1, \eta(l_1))^L) = \alpha([\text{P}](h_2, \eta(l_2))^L)$$

**Abstract non-interference** $(\eta) \text{P}(\phi \sim \llbracket \alpha \rrbracket)$:

$$\eta(l_1) = \eta(l_2) \Rightarrow \alpha([\text{P}](\phi(h_1), \eta(l_1))^L) = \alpha([\text{P}](\phi(h_2), \eta(l_2))^L)$$

**Example**: $(\text{id}) \text{P}(\text{Sign} \sim \llbracket \text{Par} \rrbracket)$

$$\text{P} = l := l \ast h^2.$$  

$\text{Sign}(h) = +, \ l = 1 \sim \text{Par}(l) = l \text{ don't know}$

$\text{Sign}(h) = -, \ l = 1 \sim \text{Par}(l) = l \text{ don't know}$

$\sim \Rightarrow$ There is not an information flow from the sign of $h$ into the parity of $l$. 

---

**Abstracting non-interference III**
Basic properties

\[ [\eta]P(\top) \]

\[ [\eta]P(\alpha) \Rightarrow \forall \beta \subseteq \eta. [\beta]P(\alpha) \]

\[ [\eta]P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. [\eta]P(\beta) \]

\[ \forall i. [\eta]P(\alpha_i) \Rightarrow [\eta]P(\bigcap_{i \in I} \alpha_i) \]
Basic properties

6. $[\eta]P(\top)

6. $[\eta]P(\alpha) \Rightarrow \forall \beta \supseteq \eta. [\beta]P(\alpha)$

6. $[\eta]P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. [\eta]P(\beta)$

6. $\forall i. [\eta]P(\alpha_i) \Rightarrow [\eta]P(\bigcap_{i \in I} \alpha_i)$

6. $(\eta)P(\phi \sim_\exists \top)$

6. $(\eta)P(\phi \sim_\exists \alpha) \Rightarrow \forall \beta \supseteq \alpha. (\eta)P(\phi \sim_\exists \beta)$

6. $\forall i. (\eta)P(\phi \sim_\exists \alpha_i) \Rightarrow (\eta)P(\phi \sim_\exists \bigcap_{i \in I} \alpha_i)$
Basic properties

6. \([\eta] P(\top)\)

6. \([\eta] P(\alpha) \Rightarrow \forall \beta \subseteq \eta. [\beta] P(\alpha)\)

6. \([\eta] P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. [\eta] P(\beta)\)

6. \(\forall i. [\eta] P(\alpha_i) \Rightarrow [\eta] P(\bigcap_{i \in I} \alpha_i)\)

6. \((\eta) P(\phi \leadsto \top)\)

6. \((\eta) P(\phi \leadsto \alpha) \Rightarrow \forall \beta \supseteq \alpha. (\eta) P(\phi \leadsto \beta)\)

6. \(\forall i. (\eta) P(\phi \leadsto \alpha_i) \Rightarrow (\eta) P(\phi \leadsto \bigcap_{i \in I} \alpha_i)\)

Standard non-interference: \([id] P(id) = (id) P(id \leadsto [id])\)
Basic properties

1. $[\eta]P(\top)$

2. $[\eta]P(\alpha) \Rightarrow \forall \beta \subseteq \eta. [\beta]P(\alpha)$

3. $[\eta]P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. [\eta]P(\beta)$

4. $\forall i. [\eta]P(\alpha_i) \Rightarrow [\eta]P(\bigcap_{i \in I} \alpha_i)$

5. $(\eta)P(\phi \leadsto \top)$

6. $(\eta)P(\phi \leadsto \alpha) \Rightarrow \forall \beta \supseteq \alpha. (\eta)P(\phi \leadsto \beta)$

7. $\forall i. (\eta)P(\phi \leadsto \alpha_i) \Rightarrow (\eta)P(\phi \leadsto \bigcap_{i \in I} \alpha_i)$

$[\text{id}]P(\text{id}) \Rightarrow (\eta)P(\text{id} \leadsto \alpha) \Rightarrow (\eta)P(\phi \leadsto \alpha)$
Basic properties

\[ \eta \] P(\top) 

\[ \eta \] P(\alpha) \Rightarrow \forall \beta \subseteq \eta. \ [\beta] P(\alpha) 

\[ \eta \] P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. \ [\eta] P(\beta) 

\forall i. \ [\eta] P(\alpha_i) \Rightarrow [\eta] P(\bigcap_{i \in I} \alpha_i) 

(\eta) P(\phi \sim \bot) 

(\eta) P(\phi \sim \top) \Rightarrow \forall \beta \supseteq \alpha. \ (\eta) P(\phi \sim \bot) 

\forall i. \ (\eta) P(\phi \sim \bot \alpha_i) \Rightarrow (\eta) P(\phi \sim \bot \bigcap_{i \in I} \alpha_i) 

[\eta] P(\alpha) \Rightarrow (\eta) P(id \sim \bot \alpha) \Rightarrow (\eta) P(\phi \sim \bot \alpha)
Basic properties

6. \([\eta]P(\top)\)

6. \([\eta]P(\alpha) \Rightarrow \forall \beta \subseteq \eta. [\beta]P(\alpha)\)

6. \([\eta]P(\alpha) \Rightarrow \forall \beta \supseteq \alpha. [\eta]P(\beta)\)

6. \(\forall i. [\eta]P(\alpha_i) \Rightarrow [\eta]P(\bigcap_{i \in I} \alpha_i)\)

6. \((\eta)P(\phi \leadsto T)\)

6. \((\eta)P(\phi \leadsto \alpha) \Rightarrow \forall \beta \supseteq \alpha. (\eta)P(\phi \leadsto \beta)\)

6. \(\forall i. (\eta)P(\phi \leadsto \alpha_i) \Rightarrow (\eta)P(\phi \leadsto \bigcap_{i \in I} \alpha_i)\)

[\text{id}]P(id) \not\Rightarrow [\eta]P(\alpha) \text{ due to deceptive flows}
Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...)  

[Giacobazzi & Ranzato ’97]

Designing abstractions = designing attackers
Deriving output attackers

Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...) [Giacobazzi & Ranzato ’97]

Designing abstractions = designing attackers

Characterize the most concrete $\alpha$ such that $(\eta)P(\phi \rightsquigarrow\alpha)$

[The most powerful output attacker]
The following theorems hold:

Consider $\eta \in Abs(\varphi(\mathbb{V}_L))$:

We characterize the function $\lambda \eta \cdot [\eta][P](id)$ whose result is

$$\prod \left\{ \beta \mid [\eta]P(\beta) \right\}.$$
The following theorems hold:

1. Consider $\eta \in \text{Abs}(\varphi(\mathcal{V}_L))$:
   We characterize the function $\lambda \eta \cdot [\eta][P](\text{id})$ whose result is
   $$\bigcap \left\{ \beta \mid \eta \mathcal{P}(\beta) \right\}.$$ 

2. Consider $\eta \in \text{Abs}(\varphi(\mathcal{V}_L))$ and $\phi \in \text{Abs}(\varphi(\mathcal{V}_H))$:
   We characterize the function $\lambda \eta \cdot (\eta)[P](\phi \leadsto \text{id})$ whose result is
   $$\bigcap \left\{ \beta \mid (\eta)\mathcal{P}(\phi \leadsto \beta) \right\}.$$
Deriving output attackers

The following theorems hold:

1. Consider $\eta \in \text{Abs}(\wp(\mathbb{V}^L))$:
   We characterize the function $\lambda \eta. \, [\eta][P](\text{id})$ whose result is
   $\prod \{ \beta \mid [\eta]P(\beta) \}.$

2. Consider $\eta \in \text{Abs}(\wp(\mathbb{V}^L))$ and $\phi \in \text{Abs}(\wp(\mathbb{V}^H))$:
   We characterize the function $\lambda \eta. \, (\eta)[P](\phi \sim \text{id})$ whose result is
   $\prod \{ \beta \mid (\eta)P(\phi \sim \beta) \}.$

$\Rightarrow$ This would provide a certificate for security with a fixed input observation.
Characterizing Attackers

Let $\eta \in uco(\varphi(V^L))$, $\phi \in uco(\varphi(V^H))$:

**Which points guarantee non-interference?**
Let $\eta \in uco(\varphi(\mathbb{V}^L))$, $\phi \in uco(\varphi(\mathbb{V}^H))$:

**Which points guarantee non-interference?**

**Indistinguishable elements:**

\[ \gamma_{[P]}^\eta(l) \overset{\text{def}}{=} \left\{ [P](h, y)^L \mid \eta(y) = \eta(l), \ h \in \mathbb{V}^H \right\}; \]

\[ \gamma_{[P]}^{\eta, \phi}(l) \overset{\text{def}}{=} \left\{ [P](\phi(h), \eta(l))^L \mid h \in \mathbb{V}^H \right\} \]
Characterizing Attackers

Let $\eta \in uco(\varphi(\mathbb{V}_L))$, $\phi \in uco(\varphi(\mathbb{V}_H))$:

**Which points guarantee non-interference?**

### Indistinguishable elements:

- $\gamma^\eta_{[\mathcal{P}]}(l) \overset{\text{def}}{=} \{ [\mathcal{P}](h, y)^L \mid \eta(y) = \eta(l), \ h \in \mathbb{V}_H \}$

- $\gamma^\eta,\phi_{[\mathcal{P}]}(l) \overset{\text{def}}{=} \{ [\mathcal{P}](\phi(h), \eta(l))^L \mid h \in \mathbb{V}_H \}$

### Secret elements:

\[
\text{Secr}^\epsilon_{[\mathcal{P}]}(X) \iff \\
\forall l \in \mathbb{V}_L . (\exists Z \in \gamma^\epsilon_{[\mathcal{P}]}(l) . Z \subseteq X \Rightarrow \forall W \in \gamma^\epsilon_{[\mathcal{P}]}(l) . W \subseteq X)
\]

Deriving and Proving Abstract Non-interference – p.12/34
Consider $X, Y \in \wp(\mathbb{V}^L)$, and $\text{Secr}^\epsilon_{[P]}$ of both:
Consider $X, Y \in \varrho(\mathcal{V}^L)$, and $\text{Secr}^\varepsilon_{[P]}$ of both:

Graphically

$\gamma^\varepsilon_{[P]}(l_1)$

$\gamma^\varepsilon_{[P]}(l_3)$
Consider $X, Y \in \wp(\mathbb{V}^L)$, and $\text{Secr}_P^\epsilon$ of both:

$$\Rightarrow \text{Secr}_P^\epsilon(X) \text{ and } -\text{Secr}_P^\epsilon(Y)$$
Consider a program $P$, $\eta \in uco(\wp(\mathbb{V}^L))$ and $\phi \in uco(\wp(\mathbb{V}^H))$

\[
\downarrow
\]

\[
S^n_P(\wp(\mathbb{V}^L)) = \left\{ X \in \wp(\mathbb{V}^L) \mid Secr^n_P(X) \right\}
\]

is the most concrete domain $\mathcal{K}$ such that $[\eta]_P(\mathcal{K})$
Consider a program $P$, $\eta \in uco(\wp(\mathbb{V}^L))$ and $\phi \in uco(\wp(\mathbb{V}^H))$

\[ \downarrow \]

\[ S^n_P(\wp(\mathbb{V}^L)) = \left\{ X \in \wp(\mathbb{V}^L) \mid \text{Secr}^n_P(X) \right\} \]

is the most concrete domain $\mathcal{K}$ such that $[\eta]P(\mathcal{K})$

\[ S^n_P,\phi(\wp(\mathbb{V}^L)) = \left\{ X \in \wp(\mathbb{V}^L) \mid \text{Secr}^n_P,\phi(X) \right\} \]

is the most concrete domain $\mathcal{K}$ such that $(\eta)P(\phi \sim \mathcal{K})$
Characterizing Attackers II

Consider a program $P$, $\eta \in uco(\wp(\mathbb{V}^L))$ and $\phi \in uco(\wp(\mathbb{V}^H))$

\[ S^n_P(\wp(\mathbb{V}^L)) = \left\{ X \in \wp(\mathbb{V}^L) \mid Secr^n_{[P]}(X) \right\} \]

is the most concrete domain $\mathcal{K}$ such that $[\eta]P(\mathcal{K})$

\[ S^n_P,\phi(\wp(\mathbb{V}^L)) = \left\{ X \in \wp(\mathbb{V}^L) \mid Secr^n_{[P]}(X) \right\} \]

is the most concrete domain $\mathcal{K}$ such that $(\eta)P(\phi \sim \mathcal{K})$

$\Rightarrow$ The canonical attacker is obtained by a fix-point process!
Let $\eta \in uco(\varphi(\mathbb{V}^L))$, $\phi \in uco(\varphi(\mathbb{V}^H))$, $P$ a program and $\eta(l_1) = \eta(l_2)$:

**Which elements are always not secret?**
Let $\eta \in uco(\varphi(\mathbb{V}^L))$, $\phi \in uco(\varphi(\mathbb{V}^H))$, $P$ a program and $\eta(l_1) = \eta(l_2)$:

**Which elements are always not secret?**

Narrow (abstract) Non-Interference
Let $\eta \in uco(\varphi(V^L))$, $\phi \in uco(\varphi(V^H))$, $P$ a program and $\eta(l_1) = \eta(l_2)$: $D_P(\eta)$. 

Narrow (abstract) Non-Interference 

$\rho([P](h_1, l_1)) = \rho([P](h_2, l_2))$
Let \( \eta \in uco(\wp(\mathbb{V}^L)) \), \( \phi \in uco(\wp(\mathbb{V}^H)) \), \( P \) a program and \( \eta(l_1) = \eta(l_2) \): \( D_P(\eta) \).
Let $\eta \in uco(\rho(\forall^L))$, $\phi \in uco(\rho(\forall^H))$, $P$ a program and $\eta(l_1) = \eta(l_2)$:

For which elements we have to verify secrecy? $\uparrow \mathcal{D}[P](\eta)$. 
Let $\eta \in uco(\phi(\mathbb{V}^L))$, $\phi \in uco(\phi(\mathbb{V}^H))$, $P$ a program and $\eta(l_1) = \eta(l_2)$:

**FOR WHICH ELEMENTS WE HAVE TO VERIFY SECRECY?**

$\uparrow\mathcal{D}[P](\eta)$. 

\[ [P](\mathbb{V}^H, \eta(l_1)) \]

\[ [P](\phi(h_1), \eta(l_1)) \]

\[ [P](\phi(h_2), \eta(l_1)) \]

\[ [P](\mathbb{V}^H, \eta(l_2)) \]

\[ [P](\phi(h_1), \eta(l_2)) \]
Graphically

Let $\eta \in uco(\varphi(\mathcal{V}^L))$, $\eta(l_1) \neq \eta(l_2)$:

$$D[p](\eta).$$
Let $\eta \in uco(\mathcal{P}(V^L))$, $\eta(l_1) \neq \eta(l_2)$: $\gamma_P^\varepsilon(l_1)$, $\gamma_P^\varepsilon(l_2)$, $\gamma_P^\varepsilon(l_3)$, $\gamma_P^\varepsilon(l_4)$.
Let $\eta \in uco(\wp(\mathcal{V}^L))$, $\eta(l_1) \neq \eta(l_2)$: $\mathcal{D}[P](\eta)$.
Let $\eta \in uco(\wp(\mathcal{V}^L))$, $\eta(l_1) \neq \eta(l_2)$: $S^\varepsilon_P(\uparrow \mathcal{D}_P(\eta))$. 

Graphically

Deriving and Proving Abstract Non-interference – p.17/34
Let $\eta \in uco(\rho(V^L))$, $\phi \in uco(\rho(V^H))$, $\phi(h_1) \neq \phi(h_2)$ and $\eta(l_1) \neq \eta(l_2)$:

**Which elements are always secret?** $\text{Irr}_p(\phi, \eta)$.
Deriving canonical attackers

Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...)  [Giacobazzi & Ranzato ’97]

Transforming abstractions = transforming attackers
Abstract interpretation provides advanced methods for designing abstractions (refinement, simplification, compression ...) [Giacobazzi & Ranzato '97]

Transforming abstractions = transforming attackers

\[\text{Characterize the most concrete } \delta \text{ such that } (\delta) P (\phi \sim_{\|} \delta)\]

[The most powerful \textit{canonical} attacker]

\[\Rightarrow \text{This would provide a certificate for security.}\]
Deriving canonical attackers

\[ \lambda X. [X][P](id) \text{ is monotone on } \text{Abs}(\varphi(V^L)). \]

\[ [\alpha]P(\alpha) \iff \alpha = [\alpha][P](id). \]

\[ \text{lfp}(\lambda X. [X][P](id)) \text{ is the most concrete secure attacker for } P \text{ for narrow abstract non-interference.} \]
Deriving canonical attackers

\[ \lambda X. \mathbb{[}X\mathbb{]}[P](\text{id}) \text{ is monotone on } Abs(\wp(\forall L)). \]

\[ [\alpha]P(\alpha) \iff \alpha = [\alpha][P](\text{id}). \]

\[ \text{lf}(\lambda X. \mathbb{[}X\mathbb{]}[P](\text{id})) \text{ is the most concrete secure attacker for } P \text{ for narrow abstract non-interference.} \]

\[ (\alpha)P(\phi \sim\text{[id]}) \iff \alpha = (\alpha)[P](\phi \sim\text{[id]}) \]

\[ \lambda X. (X)[P](\phi \sim\text{[id]}) \sqcup X \text{ is extensive on } Abs(\wp(\forall L)). \]

\[ \text{fix}(\lambda X. (X)[P](\phi \sim\text{[id]}) \sqcup X) \text{ is a secure attacker for } P \text{ for abstract non-interference.} \]
Deriving canonical attackers

**Example:**

\[
P = \textbf{while } h \textbf{ do } (l := l \times 2; \ h := h - 1)
\]

.... we derive a secure attacker \( \pi = \bigvee \left( \{ n2^{\mathbb{N}} \mid n \in 2\mathbb{N} + 1 \} \cup \{0\} \right) \):

\[
(\pi)[P](\text{id} \sim \|\pi)
\]

\[
h = 0, \ \pi(l) = 32^{\mathbb{N}} \sim \pi(l) = 32^{\mathbb{N}}
\]

\[
h = 2, \ \pi(l) = 32^{\mathbb{N}} \sim \pi(l) = 32^{\mathbb{N}}
\]

\(
\sim \) In the program \( l \) is always multiplied by \( 2! \)
Abstract robust declassification

Consider a program $P$ and its finite computations.

A passive attacker may be able to learn some information by observing the system but, by assumption, that information leakage is allowed by the security policy.

[Zdancewic and Myers 2001]
Abstract robust declassification

Consider a program $P$ and its finite computations.

A passive attacker may be able to learn some information by observing the system but, by assumption, that information leakage is allowed by the security policy.

[Zdancewic and Myers 2001]

We want to characterize the most concrete flow-irredundant property such that $(\eta)P(\phi \leadsto \alpha)$

[The maximal amount of information disclosed]

$\Rightarrow$ This would provide a certificate for disclosed secrets.
Abstract robust declassification

Consider the program

\[ P = l := l + (h \mod 3) \]

The transition system is such that \( < h, l > \mapsto < h, l + (h \mod 3) > \).

Consider \( \eta(\phi(\mathbb{Z})) = \{\mathbb{Z}, [2, 4], [5, 8], \{5\}, \emptyset\} \) and \( \alpha = \text{id} \).
Abstract robust declassification

Consider the program

\[ P = \ l := l + (h \mod 3) \]

The transition system is such that \(< h, l \mapsto < h, l + (h \mod 3) >\). Consider \( \eta(\varphi(\mathbb{Z})) = \{\mathbb{Z}, [2, 4], [5, 8], \{5\}, \emptyset\} \) and \( \alpha = \text{id} \).

The flow is revealed when \( h_1 \) and \( h_2 \) differs in the values \( h_1 \mod 3 \) and \( h_2 \mod 3 \)

\[ \Rightarrow \phi = \gamma(\{3\mathbb{Z}, 3\mathbb{Z} + 1, 3\mathbb{Z} + 2\}) \]

is the maximal amount of information disclosed!
We introduce a compositional proof-system for certifying abstract non-interference;
We introduce a compositional proof-system for certifying abstract non-interference;

**Proof-system of invariants** $\mathcal{I}$: \( \{\rho\}_L c \{\rho\}_L \) means that $c$ is $\rho$-observably equivalent to the statement $\text{nil}$:

\[
\{\rho\}_L \ c \ \{\rho\}_L \iff \rho([c]([h, l])^L) = \rho(l)
\]
We introduce a compositional proof-system for certifying abstract non-interference;

- Proof-system of invariants $\mathcal{I}$: $\{\rho\}_L c \{\rho\}_L$ means that $c$ is $\rho$-observably equivalent to the statement nil:

$$\{\rho\}_L c \{\rho\}_L \iff \rho([c](h, l)^L) = \rho(l)$$

- Proof-system for deterministic narrow non-interference $\mathcal{N}_{det}$: Syntax-driven certification of narrow non-interference for deterministic languages;

- Proof-system for non-deterministic narrow non-interference $\mathcal{N}$
We introduce a compositional proof-system for certifying abstract non-interference;

- **Proof-system of invariants \( \mathcal{I} \):** \( \{\rho\}_L \ c \ \{\rho\}_L \) means that \( c \) is \( \rho \)-observably equivalent to the statement \textit{nil}:

\[
\{\rho\}_L \ c \ \{\rho\}_L \iff \rho([c](h, l)) = \rho(l)
\]

- **Proof-system for deterministic narrow non-interference \( \mathcal{N}_{\text{det}} \):** Syntax-driven certification of narrow non-interference for deterministic languages;

- **Proof-system for non-deterministic narrow non-interference \( \mathcal{N} \):**

- **Proof-system for non-deterministic abstract non-interference \( \mathcal{A} \):** Syntax-driven certification of abstract non-interference for non-deterministic languages.
The invariants proof-system $\mathcal{I}$

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I1:</td>
<td>${T}_L \vdash {T}_L$</td>
<td></td>
</tr>
<tr>
<td>I2:</td>
<td>${\rho}_L \vdash \nil {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I3:</td>
<td>$\forall x : H \rightarrow {\rho}_L x := e {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I4:</td>
<td>${\rho} \langle e, x \rangle {\rho}, x : L \vdash {\rho}_L x := e {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I5:</td>
<td>${\rho}_L c_1 {\rho}_L, {\rho}_L c_2 {\rho}_L \vdash {\rho}_L c_1 ; c_2 {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I6:</td>
<td>${\rho}_L c {\rho}_L \vdash {\rho}_L \text{ while } x \text{ do } c \text{ endw } {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I7:</td>
<td>$\forall \rho' \leq \rho \rightarrow {\rho'}_L c {\rho'}_L, {\rho}_L \vdash {\rho}_L c {\rho}_L$</td>
<td></td>
</tr>
<tr>
<td>I8:</td>
<td>$\forall i \in I . {\rho_i}<em>L c_i {\rho_i}<em>L \vdash \bigcup</em>{i \in I} \rho_i \vdash \Box_i c_i \bigcup</em>{i \in I} \rho_i$</td>
<td></td>
</tr>
</tbody>
</table>

Next Table
The assignment

\[
\{\rho\} \langle e, x \rangle \{\rho\}, \ x : L
\]

\[
\{\rho\}_L \ x := e \{\rho\}_L
\]

where

\[
\{\rho\} \langle e, x \rangle \{\rho\} \iff \rho(\mathcal{E}[e](h, l)) = \rho(l_{|_x}).
\]
The assignment

\[ \{\rho\} \langle e, x \rangle \{\rho\}, \ x : L \]

\[ \{\rho\} \_x := e \{\rho\} \_L \]

where

\[ \{\rho\} \langle e, x \rangle \{\rho\} \iff \rho(\mathcal{E}[e](h, l)) = \rho(l|_{x}) . \]

**Example:**

Let \( e = l + 2 \). Then \( \not\equiv \{Sign\} \langle e, l \rangle \{Sign\} \) since if \( l = -1 \)

\[
Sign(l + 2) = Sign(1) = + \neq Sign(l) = -
\]
The assignment

\[
\{\rho\} \langle e, x \rangle \{\rho\}, \ x : L
\]

\[
\frac{}{\{\rho\}_L x := e \{\rho\}_L}
\]

where

\[
\{\rho\} \langle e, x \rangle \{\rho\} \iff \rho(\mathcal{E}[e](h, l)) = \rho(l_{|x}).
\]

**Example:**

Let \( e = l + 2 \). We have \( \models \{\text{Par}\} \langle e, l \rangle \{\text{Par}\} \).

Consider \( c = l := l + 2 \), we obtain that

\[
\{\text{Par}\}_L l := l + 2 \{\text{Par}\}_L
\]
The sequential composition

\[
\{\rho\}_L c_1 \{\rho\}_L, \{\rho\}_L c_2 \{\rho\}_L
\]

\[
\{\rho\}_L c_1 ; c_2 \{\rho\}_L
\]
The sequential composition

\[ \{ \rho \}_L c_1 \{ \rho \}_L, \{ \rho \}_L c_2 \{ \rho \}_L \]

\[ \{ \rho \}_L c_1 ; c_2 \{ \rho \}_L \]

**Example:**

Let \( c = l := l + 2; h := h + 1. \)

\( \Rightarrow \) \[ \models \{ Par \}_L l := l + 2 \{ Par \}_L \] by Rule I4

\[ \models \{ Par \}_L h := h + 1 \{ Par \}_L \] by Rule I3
The sequential composition

\[
\{\rho\}_L \ c_1 \ \{\rho\}_L , \ \{\rho\}_L \ c_2 \ \{\rho\}_L
\]

\[
\{\rho\}_L \ c_1 ; \ c_2 \ \{\rho\}_L
\]

**Example:**

Let \( c = l := l + 2; \ h := h + 1 \).

\[
\Rightarrow \begin{cases} 
\models \{\text{Par}\}_L \ l := l + 2 \ \{\text{Par}\}_L \text{ by Rule I4} \\
\models \{\text{Par}\}_L \ h := h + 1 \ \{\text{Par}\}_L \text{ by Rule I3} 
\end{cases}
\]

\[
\Downarrow
\]

\[
\models \{\text{Par}\}_L \ l := l + 2; \ h := h + 1 \ \{\text{Par}\}_L
\]
The proof-system $\mathcal{N}_{\text{det}}$

<table>
<thead>
<tr>
<th>Rule</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1:</td>
<td>$[\eta]<a href="%5Ctext%7Bid%7D">c</a> \subseteq \rho$</td>
</tr>
<tr>
<td>N2:</td>
<td>$[\eta]c(\top)$</td>
</tr>
<tr>
<td>N3:</td>
<td>$[\eta]\text{nil}(\rho)$</td>
</tr>
<tr>
<td>N4:</td>
<td>$[\eta]e(\rho), [\Pi(\eta) \subseteq \Pi(\rho)], \chi : L$</td>
</tr>
<tr>
<td>N5:</td>
<td>$\chi : H, \Pi(\eta) \subseteq \Pi(\rho)$</td>
</tr>
<tr>
<td>N6:</td>
<td>$[\eta]c_1(\rho), [\rho]c_2(\beta)$</td>
</tr>
<tr>
<td>N7:</td>
<td>${\rho}_L c {\rho}_L$</td>
</tr>
<tr>
<td>N8:</td>
<td>$\forall i \in I. [\eta]c(\rho_i)$</td>
</tr>
<tr>
<td>N9:</td>
<td>$\forall i \in I. [\eta]c(\rho_i)$</td>
</tr>
<tr>
<td>N10:</td>
<td>$[\eta'][c'(\rho'), \eta \subseteq \eta', \rho' \subseteq \rho]$</td>
</tr>
</tbody>
</table>
The low assignment

\[ [\eta]e(\rho), \ [\Pi(\eta) \subseteq \Pi(\rho)], \ x : L \]

\[ \vdash [\eta]x := e(\rho) \]

where

\[ [\eta]e(\rho) \iff \rho(\mathcal{E}[e](h_1, l_1)) = \rho(\mathcal{E}[e](h_2, l_2)). \]
The low assignment

$$\llbracket \eta \rrbracket e(\rho), \ [\Pi(\eta) \subseteq \Pi(\rho)], \ x : \mathcal{L}$$

$$\llbracket \eta \rrbracket x := e(\rho)$$

where

$$\llbracket \eta \rrbracket e(\rho) \text{ iff } \rho(\mathcal{E}[e](h_1, l_1)) = \rho(\mathcal{E}[e](h_2, l_2)).$$

**Example:**

Let $c = l_1 := 2 * h * l_2$.
Then $\not\models [\top] l_1 := 2 * h * l_2 (\textit{Par})$ since

$$\text{Par}(\llbracket l_1 := 2 * h * l_2 \rrbracket(h, \langle l_1, 3 \rangle)^L) = \langle \text{even, odd} \rangle \neq \langle \text{even, even} \rangle$$

This because $\top(2) = \top(3)$ but $\textit{Par}(2) \neq \textit{Par}(3)$. 
The low assignment

\[ [\eta]e(\rho), [\Pi(\eta) \subseteq \Pi(\rho)], x : L \]

\[ \underbrace{x := e(\rho)} \]

where

\[ [\eta]e(\rho) \text{ iff } \rho(\mathcal{E}[e](h_1, l_1)) = \rho(\mathcal{E}[e](h_2, l_2)). \]

**NOTE:** If there’s only one low variable the condition \( \Pi(\eta) \subseteq \Pi(\rho) \) is not necessary.

**EXAMPLE:**

Consider \( c = l := 2 * h \)

\[ [\top]2 * h(Par) \quad \Rightarrow \quad [\top]l := 2 * h(Par) \]
The high assignment

\( x : \mathcal{H}, \quad \Pi(\eta) \sqsubseteq \Pi(\rho) \)

\[ [\eta]x := e(\rho) \]
The high assignment

\[ x : H, \ \Pi(\eta) \subseteq \Pi(\rho) \]

\[ [\eta]x := e(\rho) \]

**Example:**

Let \( c = h := h + 1 \). Then

\[ \rho([h := h + 1](h_1, l_1)^L) = \rho(l_1) \]
\[ \rho([h := h + 1](h_2, l_2)^L) = \rho(l_2) \]

Therefore

\[ [\eta]h := h + 1(\rho) \iff (\eta(l_1) = \eta(l_2) \Rightarrow \rho(l_1) = \rho(l_2)) \]
\[ \iff \Pi(\eta) \subseteq \Pi(\rho) \]
The proof-system $\mathcal{N}$

\[
\begin{align*}
\text{N1: } & \quad \begin{array}{l}
[\eta][c](\text{id}) \subseteq \rho \\
\hline
[\eta]c(\rho)
\end{array} \\
\text{N2: } & \quad [\eta]c(\top) \\
\text{N3: } & \quad [\eta]\text{nil}(\rho) \\
\text{N4: } & \quad [\eta]e(\rho), \ [\Pi(\eta) \subseteq \Pi(\rho)], \ x : L \\
& \quad [\eta]x := e(\rho) \\
\text{N5: } & \quad x : H, \ \Pi(\eta) \subseteq \Pi(\rho) \\
& \quad [\eta]x := e(\rho) \\
\text{N6: } & \quad [\eta]c_1(\gamma(\rho)), \ [\rho]c_2(\gamma(\beta)) \\
& \quad [\eta]c_1; c_2(\gamma(\beta)) \\
\text{N7: } & \quad \{\rho\}_L \ c \ \{\rho\}_L \\
& \quad [\rho]\text{while } x \ \text{do } c \ \text{endw}(\rho) \\
\text{N8: } & \quad \forall i \in I. \ [\eta]c(\rho_i) \\
& \quad [\eta]c(\bigcup_{i \in I} \rho_i) \\
\text{N9: } & \quad \forall i \in I. \ [\eta]c(\rho_i) \\
& \quad [\eta]c(\bigcap_{i \in I} \rho_i) \\
\text{N10: } & \quad [\eta'][c(\rho'), \ \eta \subseteq \eta', \ \rho' \subseteq \rho] \\
& \quad [\eta]c(\rho) \\
\text{N11: } & \quad \forall i \in I. \ [\eta_i]c_i(\rho_i) \\
& \quad [\bigcap_{i \in I} \eta_i] \square_i c_i(\bigcup_{i \in I} \rho_i)
\end{align*}
\]
The non-deterministic concatenation

\[ \eta c_1(\gamma(\rho)), \rho c_2(\gamma(\beta)) \]

\[ \eta c_1 ; c_2(\gamma(\beta)) \]
The non-deterministic concatenation

\[
\eta c_1(\gamma(\rho)), \rho c_2(\gamma(\beta)) \quad \frac{}{\eta c_1; c_2(\gamma(\beta))}
\]

**Example:**

Let \( \rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\} \) and

\[
c = c_1; c_2 = \begin{cases} 
    l := 1 - (h \mod 2) \quad & \square \quad l := 2 \times (h \mod 2) + 2 \times (1 - (h \mod 2)); \\
    l := (l \mod 2) \times 4h + (1 - (l \mod 2)) \times (4h + 1)
\end{cases}
\]

then

\[
\begin{align*}
    h \in 2\mathbb{Z} & \Rightarrow \rho([c_1](h, l)^L) = \rho([1, 2]) = \mathbb{Z} \\
    h \in 2\mathbb{Z} + 1 & \Rightarrow \rho([c_1](h, l)^L) = \rho([0, 2]) = \mathbb{Z}
\end{align*}
\]

\[\Rightarrow \top c_1(\rho)\]
The non-deterministic concatenation

\[
\{\gamma(\rho)\}, \{\gamma(\beta)\}
\]

\[
\{\gamma(\beta)\}
\]

**Example:**

Let \( \rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\} \) and

\[
c = c_1; c_2 = \begin{cases} 
  l := 1 - (h \bmod 2) & \square \\
  l := 2 \times (h \bmod 2) + 2 \times (1 - (h \bmod 2)) & \\
  l := (l \bmod 2) \times 4h + (1 - (l \bmod 2)) \times (4h + 1)
\end{cases}
\]

\( [\top]c_1(\rho) \) and

\[
l \in 2\mathbb{Z} \Rightarrow \rho([c_2](h, l)_L) = \rho(\{4h + 1\}) = 4\mathbb{Z} + 1
\]

\[
l \in 2\mathbb{Z} + 1 \Rightarrow \rho([c_2](h, l)_L) = \rho(\{4h\}) = 4\mathbb{Z}
\]

\[\Rightarrow [\rho]c_2(\rho)\]
The non-deterministic concatenation

\[
[\eta]c_1(\gamma(\rho)), [\rho]c_2(\gamma(\beta)) \\
\underline{[\eta]c_1; c_2(\gamma(\beta))}
\]

**Example:**

Let \( \rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\} \) and

\[
c = c_1; c_2 = \left\lfloor \begin{array}{l}
  l := 1 - (h \mod 2) \\
  l := 2 \times (h \mod 2) + 2 \times (1 - (h \mod 2)) \\
  l := (l \mod 2) \times 4h + (1 - (l \mod 2)) \times (4h + 1)
\end{array} \right. 
\]

\([\top]c_1(\rho) \text{ and } [\rho]c_2(\rho), \text{ but} \)

\[
h \in 2\mathbb{Z} \Rightarrow \rho(\llbracket c_1; c_2 \rrbracket(h, l)_{\top}) = \rho(\{4h, 4h + 1\}) = \mathbb{Z}
\]

\[
h \in 2\mathbb{Z} + 1 \Rightarrow \rho(\llbracket c_1; c_2 \rrbracket(h, l)_{\top}) = \rho(\{4h + 1\}) = 4\mathbb{Z} + 1
\]

\[\Rightarrow \not\in [\top]c(\rho)\]
The proof-system $\mathcal{A}$

<table>
<thead>
<tr>
<th>A1:</th>
<th>(η)<a href="%5Ctext%7Bid%7D">c</a> ⊆ ρ</th>
<th>(η)c(ρ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A4:</td>
<td>(η)e(ρ), χ : L</td>
<td>(η)x := e(ρ)</td>
</tr>
<tr>
<td>A5:</td>
<td>x : H</td>
<td>(η)x := e(ρ)</td>
</tr>
<tr>
<td>A6:</td>
<td>(η)c$_1$(γ(ρ)), [ρ]c$_2$(γ(β))</td>
<td>(η)c$_1$;c$_2$(γ(β))</td>
</tr>
<tr>
<td>A7:</td>
<td>{ρ}_L c {ρ}_L, χ : H</td>
<td>(ρ)while x do c endw(ρ)</td>
</tr>
<tr>
<td>A8:</td>
<td>(ρ)c(ρ), χ : L</td>
<td>(ρ)while x do c endw(ρ)</td>
</tr>
<tr>
<td>A9:</td>
<td>∀i ∈ I . ηc$_i$(ρ$_i$)</td>
<td>(η)□$_i$c$<em>i$(∪$</em>{i∈I}$ρ$_i$)</td>
</tr>
<tr>
<td>A10:</td>
<td>(η)c(ρ'), ρ' ⊆ ρ</td>
<td>(η)c(ρ)</td>
</tr>
<tr>
<td>A11:</td>
<td>∀i ∈ I . ηc(ρ$_i$)</td>
<td>(η)c(∪$_{i∈I}$ρ$_i$)</td>
</tr>
<tr>
<td>A12:</td>
<td>∀i ∈ I . ηc(ρ$_i$)</td>
<td>(η)c(∩$_{i∈I}$ρ$_i$)</td>
</tr>
</tbody>
</table>
The high assignment

\[
\begin{align*}
\text{x : H} \\
\hline
(\eta)\text{x := e(\rho)}
\end{align*}
\]
The high assignment

\[ x : \mathcal{H} \]

\[
(\eta)x := e(\rho)
\]

**Example:**

Let \( c = h := h + 1 \). Then

\[
\rho([h := h + 1](h_1, \eta(l_1))^L) = \rho(\eta(l_1))
\]

\[
\rho([h := h + 1](h_2, \eta(l_2))^L) = \rho(\eta(l_2))
\]

Therefore

\[
[h]h := h + 1(\rho) \iff (\eta(l_1) = \eta(l_2) \implies \rho(\eta(l_1)) = \rho(\eta(l_2)))
\]
The concatenation

\[(\eta)c_1(\gamma(\rho)), [\rho]c_2(\gamma(\beta))\]

\[(\eta)c_1; c_2(\gamma(\beta))\]
The concatenation

\[(\eta)c_1(\gamma(\rho)), [\rho]c_2(\gamma(\beta))\]

\[= (\eta)c_1; c_2(\gamma(\beta))\]

**Example:**

Let \(\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\}\) and

\[c = c_1; c_2 = \begin{cases} 
  l := (h \mod 2)(2l \mod 4) + (1 - (h \mod 2))(l \mod 2 + 1); \\
  l := (l \mod 2) \times 4h + (1 - (l \mod 2)) \times (4h + 1)
\end{cases}\]

then

\[h \in 2\mathbb{Z} \implies \rho(\llbracket c_1 \rrbracket (h, \mathbb{Z}^L)) = \rho(\{1, 2\}) = \mathbb{Z}\]

\[h \in 2\mathbb{Z} + 1 \implies \rho(\llbracket c_1 \rrbracket (h, \mathbb{Z}^L)) = \rho(\{0, 2\}) = \mathbb{Z}\]

\[\implies (\top)c_1(\rho)\]
The concatenation

\[(\eta) c_1 (\gamma (\rho)), [\rho] c_2 (\gamma (\beta))\]

\[\quad (\eta) c_1 ; c_2 (\gamma (\beta))\]

**Example:**

Let \(\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\}\) and

\[c = c_1; c_2 = \begin{cases}   l := (h \mod 2)(2l \mod 4) + (1 - (h \mod 2))(l \mod 2 + 1); \\   l := (l \mod 2)*4h + (1 - (l \mod 2)) * (4h + 1) \end{cases}\]

\[(\top) c_1 (\rho)\] and

\[l \in 2\mathbb{Z} \Rightarrow \rho([c_2](h, l)^{L}) = \rho([4h + 1]) = 4\mathbb{Z} + 1\]

\[l \in 2\mathbb{Z} + 1 \Rightarrow \rho([c_2](h, l)^{L}) = \rho([4h]) = 4\mathbb{Z}\]

\[\Rightarrow [\rho] c_2 (\rho)\]
The concatenation

\[(\eta)c_1(\gamma(\rho)), [\rho]c_2(\gamma(\beta))\]

\[(\eta)c_1; c_2(\gamma(\beta))\]

**Example:**

Let \(\rho = \{\mathbb{Z}, 4\mathbb{Z}, 4\mathbb{Z} + 1, 4\mathbb{Z} + 2, 4\mathbb{Z} + 3, \emptyset\}\) and

\[
c = c_1; c_2 = \begin{cases} 
  l := (h \mod 2)(2l \mod 4) + (1 - (h \mod 2))(l \mod 2 + 1); \\
  l := (l \mod 2) \times 4h + (1 - (l \mod 2)) \times (4h + 1)
\end{cases}
\]

*(\top)c_1(\rho) and [\rho]c_2(\rho)*, but

\[
h \in 2\mathbb{Z} \implies \rho([c_1; c_2](h, Z^L)) = \rho(\{4h, 4h + 1\}) = \mathbb{Z}
\]

\[
h \in 2\mathbb{Z} + 1 \implies \rho([c_1; c_2](h, Z^L)) = \rho(\{4h + 1\}) = 4\mathbb{Z} + 1
\]

\[\implies \nexists (\top)c(\rho)\]
The system $\mathcal{I}$ is correct.
\textbf{T1)} The system $\mathcal{I}$ is correct.

\textbf{T2)} The system $\mathcal{N}_{\text{det}}$ is complete but $\mathcal{N}_{\text{det}} \setminus \{\text{N1}\}$ is correct.

\textbf{Example:}

Let $\rho = \{2\mathbb{Z}\} \cup \left\{ \{n\} \mid n \in 2\mathbb{Z} + 1 \right\}$ and

\[ P = \mathit{l} := 4 \times h^2 + 4; \ c \]

where $c = \text{while } h \text{ do } \mathit{l} := \mathit{l} \mod 4; \ h := 0 \text{ endw}$.

We have $[\top]l := 4 \times h^2 + 4(\rho) \text{ and } [\top]P(\rho)$

But $\not\models [\rho]\text{while } h \text{ do } \mathit{l} := \mathit{l} \mod 4; \ h := 0 \text{ endw}(\rho)$:

\[ \rho(\llbracket c \rrbracket(0,5]^L) = 5 \neq \rho(\llbracket c \rrbracket(1,5]^L) = 1 \]
\textbf{Theorems}

\textbf{T1)} The system $\mathcal{I}$ is correct.

\textbf{T2)} The system $\mathcal{N}_{\text{det}}$ is complete but $\mathcal{N}_{\text{det}} \setminus \{N1\}$ is correct.

\textbf{T3)} The system $\mathcal{N}$ is complete but $\mathcal{N} \setminus \{N1\}$ is correct.
Theorems

\[ T1 \] The system \( \mathcal{I} \) is correct.

\[ T2 \] The system \( \mathcal{N}_{\text{det}} \) is complete but \( \mathcal{N}_{\text{det}} \setminus \{N1\} \) is correct.

\[ T3 \] The system \( \mathcal{N} \) is complete but \( \mathcal{N} \setminus \{N1\} \) is correct.

\[ T4 \] The system \( \mathcal{A} \) is complete but \( \mathcal{A} \setminus \{A1\} \) is correct.

**Example:**

Let \( \rho = \{\mathbb{Z}, 2\mathbb{Z}, 4\mathbb{Z}, \emptyset\} \) and

\[
P = \textbf{while } h \textbf{ do } l := (l \mod 4) \ast (l \div 4); \ h := 0 \textbf{ endw}
\]

Then \( (\rho)P(\rho) \):

\[
\rho(\llbracket P \rrbracket(h, 2\mathbb{Z})^L) = 2\mathbb{Z}
\]

But \( \not\models \{\rho\}_L \ P \ {\rho\}_L \):

\[
\rho(\llbracket c \rrbracket(1, 2)^L) = \rho(0) = 4\mathbb{Z} \neq \rho(2) = 2\mathbb{Z}
\]
Theorems

\( T_1 \)  The system \( \mathcal{I} \) is correct.

\( T_2 \)  The system \( \mathcal{N}_{\text{det}} \) is complete but \( \mathcal{N}_{\text{det}} \setminus \{N1\} \) is correct.

\( T_3 \)  The system \( \mathcal{N} \) is complete but \( \mathcal{N} \setminus \{N1\} \) is correct.

\( T_4 \)  The system \( \mathcal{A} \) is complete but \( \mathcal{A} \setminus \{A1\} \) is correct.

\( T_5 \)  The system \( \mathcal{N} \) is stronger than \( \mathcal{A} \).

**Example:**

\[
P = \ h := h + 1; \ l := 2 \ast h
\]

Then \( \models [\text{Sign}]P(\text{Par}) \) but \( \not\models_{\mathcal{N}} [\text{Sign}]P(\text{Par}) : \)

\[
\text{Sign}(2) = \text{Sign}(3) \quad \text{and} \quad \text{Par}(\llbracket h := h + 1 \rrbracket (h, 3)^L) = \text{Par}(3) = \text{odd} \neq \text{Par}(\llbracket h := h + 1 \rrbracket (h, 2)^L) = \text{Par}(2) = \text{even}
\]
Theorems

\( T_1 \)  The system \( \mathcal{I} \) is correct.

\( T_2 \)  The system \( \mathcal{N}_{\text{det}} \) is complete but \( \mathcal{N}_{\text{det}} \setminus \{N1\} \) is correct.

\( T_3 \)  The system \( \mathcal{N} \) is complete but \( \mathcal{N} \setminus \{N1\} \) is correct.

\( T_4 \)  The system \( \mathcal{A} \) is complete but \( \mathcal{A} \setminus \{A1\} \) is correct.

\( T_5 \)  The system \( \mathcal{N} \) is stronger than \( \mathcal{A} \).

Example:

\[
P = \begin{cases} 
  h := h + 1; & l := 2 \times h 
\end{cases}
\]

Then \( \models \lbrack \text{Sign} \rbrack P(\text{Par}) \) but \( \nvdash_{\mathcal{N}} \lbrack \text{Sign} \rbrack P(\text{Par}) \) while \( \vdash_{\mathcal{A}} \lbrack \text{Sign} \rbrack P(\text{Par}) \)
Discussion

We map security of programs into the lattice of abstract interpretations:

- systematic methods for designing attackers and certificates
- security degrees compared in the lattice
- checking abstract non-interference by static program analysis
We map security of programs into the lattice of abstract interpretations:

- systematic methods for designing attackers and certificates
- security degrees compared in the lattice
- checking abstract non-interference by static program analysis

Abstract non-interference is a semantics property

- the method is language independent (as any abstract interpretation)
- refined semantics may refine security:
  covert channels (termination, non-determinism, synchronization, probabilistic, etc...) is a matter of semantics!
Discussion

We map security of programs into the lattice of abstract interpretations:
- systematic methods for designing attackers and certificates
- security degrees compared in the lattice
- checking abstract non-interference by static program analysis

Abstract non-interference is a semantics property
- the method is language independent (as any abstract interpretation)
- refined semantics may refine security:
  covert channels (termination, non-determinism, synchronization, probabilistic, etc...) is a matter of semantics!

How far is any practical application?
- program slicing may help in checking program secrecy!
- Construction of a PCC architecture.