On the Decidability of Model-Checking Information Flow Properties

Raghavendra K. R.
Joint Work: Deepak D'Souza, Janardhan Kulkarni, Barbara Sprick, Raveendra Holla
Indian Institute of Science, Bangalore

Introduction to Software Security

Protecting the confidentiality of information manipulated by computing systems is a long standing yet increasingly important problem. There is little assurance that current computing systems protect data confidentiality and integrity. - Myers (FM for Security)

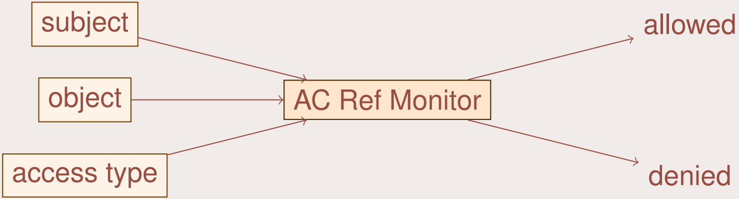
Access Control

Limitation: Does NOT address end-to-end security.

Introduction to Software Security

Protecting the confidentiality of information manipulated by computing systems is a long standing yet increasingly important problem. There is little assurance that current computing systems protect data confidentiality and integrity. - Myers (FM for Security)

Access Control



Limitation: Does NOT address end-to-end security.

Noninterference [GM82]

- Addresses end-to-end security.
- $M = (Q, S, I, O, \delta, o, s_0)$
  $\delta : Q \times (S \times I) \to Q$,
  $o : Q \times S \to O$.

S_1 noninterferes with S_2

for all $s \in S_2$, $o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, purge_{S_1}(w)), s)$.

Noninterference [GM82]

- Addresses end-to-end security.
- $M = (Q, S, I, O, \delta, o, s_0)$
  $\delta : Q \times (S \times I) \rightarrow Q$,
  $o : Q \times S \rightarrow O$.

S_1 noninterferes with S_2

for all $s \in S_2$, $o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, purge_{S_1}(w)), s)$.

Verifying Noninterference = Reachability Check

$$M_{S_1 S_2} = (Q \times Q, S, I, O, \delta', o', (s_0, s_0))$$

$$\delta'((t_1, t_2), (s, a)) = \begin{cases} (\delta(t_1, (s, a)), t_2) & \text{if } s \in S_1 \\ (\delta(t_1, (s, a)), \delta(t_2, (s, a))) & \text{otherwise} \end{cases}$$

$$o'((t_1, t_2), s) = (o(t_1, s), o(t_2, s))$$

$M \models NI$ w.r.t S_1, S_2 [MZ07]

iff for all reachable states (t_1, t_2) of $M_{S_1 S_2}$,
$o'((t_1, t_2), s) = (o_1, o_2) \Rightarrow o_1 = o_2$ for all $s \in S_2$.

Decidable for finite state systems.

Verifying Noninterference = Reachability Check

$$M_{S_1 S_2} = (Q \times Q, S, I, O, \delta', o', (s_0, s_0))$$

$$\delta'((t_1, t_2), (s, a)) = \begin{cases} (\delta(t_1, (s, a)), t_2) & \text{if } s \in S_1 \\ (\delta(t_1, (s, a)), \delta(t_2, (s, a))) & \text{otherwise} \end{cases}$$

$$o'((t_1, t_2), s) = (o(t_1, s), o(t_2, s))$$

$M \models NI$ w.r.t S_1, S_2 [MZ07]

iff for all reachable states (t_1, t_2) of $M_{S_1 S_2}$,
$o'((t_1, t_2), s) = (o_1, o_2) \Rightarrow o_1 = o_2$ for all $s \in S_2$.

Decidable for finite state systems.

Generalized Noninterference - GNI

Limitation: Non-determinism for interrupts and concurrency.

McCullough'87

$S_1 \not\rightsquigarrow_{GNI} S_2$ iff $\forall s \in S_2\ \forall w \in (S \times I)^*\ \forall c \in (S_1 \times I)$,
$o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, w \cdot c), s)$.

- Event Systems: (E, I, O, L)
  $I, O \subseteq E, I \cap O = \emptyset, L \subseteq E^*$.
- Assume security levels: $L \leq H$.
- $\forall t_1, t_2, t_3 \in E^*$,
  $((t_1.t_2 \in L \wedge t_3 \restriction_{E \setminus (H \cap I)} = t_2 \restriction_{E \setminus (H \cap I)}) \Rightarrow$
  $\exists t_4 \in E^*.(t_1.t_4 \in L \wedge t_4 \restriction_{L \cup (H \cap I)} = t_3 \restriction_{L \cup (H \cap I)})$

Generalized Noninterference - GNI

Limitation: Non-determinism for interrupts and concurrency.

McCullough'87

$S_1 \not\rightsquigarrow_{GNI} S_2$ iff $\forall s \in S_2\ \forall w \in (S \times I)^*\ \forall c \in (S_1 \times I)$,
$o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, w \cdot c), s)$.

- Event Systems: (E, I, O, L)
  $I, O \subseteq E, I \cap O = \emptyset, L \subseteq E^*$.
- Assume security levels: $L \leq H$.
- $\forall t_1, t_2, t_3 \in E^*$,
  $((t_1.t_2 \in L \wedge t_3 \upharpoonright_{E \setminus (H \cap I)} = t_2 \upharpoonright_{E \setminus (H \cap I)}) \Rightarrow$
  $\exists t_4 \in E^*.(t_1.t_4 \in L \wedge t_4 \upharpoonright_{L \cup (H \cap I)} = t_3 \upharpoonright_{L \cup (H \cap I)})$

Generalized Noninterference - GNI

Limitation: Non-determinism for interrupts and concurrency.

McCullough'87

$S_1 \not\rightarrowtail_{GNI} S_2$ iff $\forall s \in S_2\ \forall w \in (S \times I)^*\ \forall c \in (S_1 \times I)$,
$o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, w \cdot c), s)$.

- Event Systems: (E, I, O, L)
  $I, O \subseteq E, I \cap O = \emptyset, L \subseteq E^*$.
- Assume security levels: $L \leq H$.
- $\forall t_1, t_2, t_3 \in E^*$,
  $((t_1.t_2 \in L \wedge t_3 \restriction_{E \setminus (H \cap I)} = t_2 \restriction_{E \setminus (H \cap I)}) \Rightarrow$
  $\exists t_4 \in E^*.(t_1.t_4 \in L \wedge t_4 \restriction_{L \cup (H \cap I)} = t_3 \restriction_{L \cup (H \cap I)})$

Generalized Noninterference - GNI

Limitation: Non-determinism for interrupts and concurrency.

McCullough'87

$S_1 \not\rightsquigarrow_{GNI} S_2$ iff $\forall s \in S_2\ \forall w \in (S \times I)^*\ \forall c \in (S_1 \times I)$, $o(\hat{\delta}(s_0, w), s) = o(\hat{\delta}(s_0, w \cdot c), s)$.

- Event Systems: (E, I, O, L)
  $I, O \subseteq E, I \cap O = \emptyset, L \subseteq E^*$.
- Assume security levels: $L \leq H$.
- $\forall t_1, t_2, t_3 \in E^*$,
  $((t_1.t_2 \in L \wedge t_3 \restriction_{E \setminus (H \cap I)} = t_2 \restriction_{E \setminus (H \cap I)}) \Rightarrow$
  $\exists t_4 \in E^*.(t_1.t_4 \in L \wedge t_4 \restriction_{L \cup (H \cap I)} = t_3 \restriction_{L \cup (H \cap I)})$

Variants of Noninterference

Noninference (NF) [ZL97]

$\forall t \in L,\ t \upharpoonright_L \in L.$

Separability (SEP) [McL94]

$\forall \tau, \tau' \in L,\ \ interleaving(\tau \upharpoonright_H, \tau' \upharpoonright_L) \subseteq L.$

Non Deducibility for $UI \subseteq I$ (NDO) [GN88]

$\forall t_1, t_2 \in L, \forall t \in E^*,$

$(t \upharpoonright_L = t_1 \upharpoonright_L \wedge t \upharpoonright_{H \cup (L \cap UI)} = t_2 \upharpoonright_{H \cup (L \cap UI)}) \Rightarrow t \in L.$

$\vdots$

Variants of Noninterference

Noninference (NF) [ZL97]

$\forall t \in L,\ t\restriction_L \in L.$

Separability (SEP) [McL94]

$\forall \tau, \tau' \in L,\quad interleaving(\tau\restriction_H, \tau'\restriction_L) \subseteq L.$

Non Deducibility for $UI \subseteq I$ (NDO) [GN88]

$\forall t_1, t_2 \in L, \forall t \in E^*,$

$(t\restriction_L = t_1\restriction_L \wedge t\restriction_{H\cup(L\cap UI)} = t_2\restriction_{H\cup(L\cap UI)}) \Rightarrow t \in L.$

⋮

Variants of Noninterference

Noninference (NF) [ZL97]

$\forall t \in L,\ t \restriction_L \in L.$

Separability (SEP) [McL94]

$\forall \tau, \tau' \in L,\quad \mathit{interleaving}(\tau \restriction_H, \tau' \restriction_L) \subseteq L.$

Non Deducibility for $UI \subseteq I$ (NDO) [GN88]

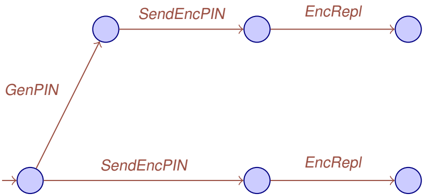
$\forall t_1, t_2 \in L, \forall t \in E^*,$

$(t \restriction_L = t_1 \restriction_L \wedge t \restriction_{H \cup (L \cap UI)} = t_2 \restriction_{H \cup (L \cap UI)}) \Rightarrow t \in L.$

$\vdots$

An example

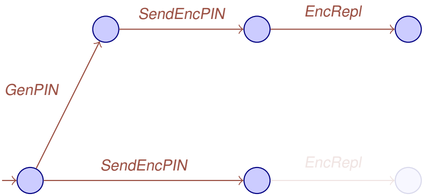
Alice wants to change her PIN.



Noninference holds. Noninference violated.

An example

Alice wants to change her PIN.



Noninference violated.

Basic Security Predicates (BSPs) [Mantel'00]

- BSP w.r.t a view = (V, N, C).
- BSP R

$$\forall \tau \in L, \Rightarrow \exists \tau',\ \tau' \restriction_C = \epsilon \wedge \tau \restriction_V = \tau' \restriction_V$$

- BSP D

$$\forall c \in C,$$
$$\forall \alpha c \beta \in L\ \wedge \beta \restriction_C = \epsilon \Rightarrow \exists \alpha' \beta',\ \alpha' \beta' \in L,\ \wedge\ \alpha =_N \alpha'\ \wedge \beta =_N \beta'$$

- BSP I

$$\forall c \in C,$$
$$\forall \alpha \beta \in L \Rightarrow \exists \alpha' \beta'\ \alpha' c \beta' \in L\ \wedge\ \alpha =_N \alpha'\ \wedge\ \beta =_N \beta'.$$

⋮

13 BSPs

Information Flow Properties and BSPs

- Let $\mathcal{H} = (L, \emptyset, H)$, and $\mathcal{HI} = (L, H \setminus I, H \cap I)$.
- $GNI(E) \Leftrightarrow BSD_{\mathcal{HI}}(E) \wedge BSI_{\mathcal{HI}}(E)$.
- $NDO(E) \Leftrightarrow BSD_{\mathcal{H}}(E) \wedge BSIA^{UI}_{\mathcal{H}}(E)$.
- $NF(E) \Leftrightarrow R_{\mathcal{H}}(E)$.
- $SEP(E) \Leftrightarrow BSD_{\mathcal{H}}(E) \wedge BSIA^{C}_{\mathcal{H}}(E)$.

$\vdots$

Model Checking BSPs

- For finite state systems, decidable [DKS'05].
- For pushdown systems (PDS), undecidable.
- Information flow properties for PDS, undecidable [To be submitted]

Undecidability for PDS

Recall $NF(E) \Leftrightarrow R_{\mathcal{H}}$.

Emptiness Problem of Turing Machines to PDS satisfying NF.

Configuration sequence is encoded on $\{v_1, v_2\}$.

Given a TM M, let L be the prefix closure of $L_1 \cup L_2$

$$L_1 = \{c \cdot enc(\#x_1\#x_2 \cdots x_n\#) \mid x_1 \text{ is a starting configuration, } x_n \text{ is an accepting configuration}\}$$

$$L_2 = \{enc(\#x_1\#x_2 \cdots x_n\#) \mid x_1 \text{ is a starting configuration, } x_n \text{ is an accepting configuration, exists } i : x_i \rightsquigarrow x_{i+1} \text{ invalid transition}\}$$

L satisfies NF iff $L(M) = \emptyset$.

Undecidability for PDS

Recall $NF(E) \Leftrightarrow R_{\mathcal{H}}$.

Emptiness Problem of Turing Machines to PDS satisfying NF.

Configuration sequence is encoded on $\{v_1, v_2\}$.

Given a TM M, let L be the prefix closure of $L_1 \cup L_2$

$$L_1 = \{c \cdot enc(\#x_1\#x_2 \cdots x_n\#) \mid \begin{array}{l} x_1 \text{ is a starting configuration,} \\ x_n \text{ is an accepting configuration}\} \end{array}$$

$$L_2 = \{enc(\#x_1\#x_2 \cdots x_n\#) \mid \begin{array}{l} x_1 \text{ is a starting configuration,} \\ x_n \text{ is an accepting configuration,} \\ \text{exists } i : x_i \rightsquigarrow x_{i+1} \text{ invalid transition}\} \end{array}$$

L satisfies NF iff $L(M) = \emptyset$.

Undecidability for PDS

Recall $NF(E) \Leftrightarrow R_{\mathcal{H}}$.

Emptiness Problem of Turing Machines to PDS satisfying NF.

Configuration sequence is encoded on $\{v_1, v_2\}$.

Given a TM M, let L be the prefix closure of $L_1 \cup L_2$

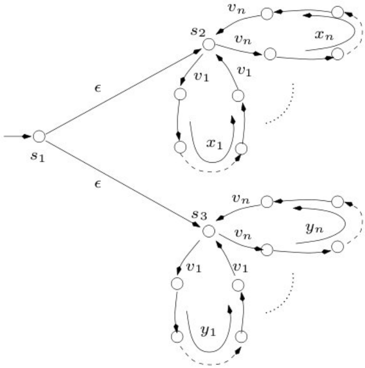
$$L_1 = \{c \cdot enc(\#x_1\#x_2 \cdots x_n\#) \mid \begin{array}{l} x_1 \text{ is a starting configuration,} \\ x_n \text{ is an accepting configuration}\} \end{array}$$

$$L_2 = \{enc(\#x_1\#x_2 \cdots x_n\#) \mid \begin{array}{l} x_1 \text{ is a starting configuration,} \\ x_n \text{ is an accepting configuration,} \\ \text{exists } i : x_i \rightsquigarrow x_{i+1} \text{ invalid transition}\} \end{array}$$

L satisfies NF iff $L(M) = \emptyset$.

Weak Non-Inference

- WNI
  $\forall \tau \in L,\ \tau \restriction_C = \epsilon \Rightarrow \exists \tau' \in L,\ \tau \restriction_V = \tau' \restriction_V \ \wedge\ \tau \restriction_C \neq \tau' \restriction_C.$
- Undecidable for finite state systems.
- PCP has a solution iff T_P does not satisfy WNI.



Logic Characterization for BSPs

- WNI not definable with BSPs.
- $|V| = |C| = 1$: decidable for PDS - reduced to PA
- Natural: $FO_{=}(\cdot, \upharpoonright)$ is undecidable.

Summary and Future Work

- Model-Checking noninterference and its variants for PDS is undecidable.
- Attempted logic characterization for BSPs is undecidable.

For Future

- Decidable Logic characterization of noninterference and its variants.
- Static / Dynamic analysis of programs for refined noninterference.

Thank You