Process Opacity and Insertion Functions

Damas P. Gruska M. Carmen Ruiz

Comenius University, Slovakia

CS&P'21, 27th September, 2021

Outline

Security Properties based on Information Flow

Information Flow:

Private activities or data can influence public ones.

Security Properties based on Information Flow

Information Flow:

Private activities or data can influence public ones.

Information Flow based Attack:

An intruder tries to deduce a private property over system activities, or data by means of information flow.

Security Properties based on Information Flow

Information Flow:
Private activities or data can influence public ones.

Information Flow based Attack:
An intruder tries to deduce a private property over system activities, or data by means of information flow.

Security based on Information Flow:
Systems are considered to be secure if there is no information flow between private and public activities or data.

Security Properties

Differences:

Security Properties

Differences:

- which kind of activities or data are considered to by private,

Security Properties

Differences:

- which kind of activities or data are considered to by private,

- which kind of activities or data are considered to by public,

Security Properties

Differences:

- which kind of activities or data are considered to by private,

- which kind of activities or data are considered to by public,

- nature of attackers observations (bisimulation, . . . , traces, . . .),

Security Properties

Differences:

- which kind of activities or data are considered to by private,

- which kind of activities or data are considered to by public,

- nature of attackers observations (bisimulation, . . . , traces, . . .),

- other capabilities of attackers (time, statistic distributions, prebelief, . . .),

Security Properties

Differences:

- which kind of activities or data are considered to by private,

- which kind of activities or data are considered to by public,

- nature of attackers observations (bisimulation, . . . , traces, . . .),

- other capabilities of attackers (time, statistic distributions, prebelief, . . .),

- qualitative vs. quantitative properties.

Security Properties

Differences:

- which kind of activities or data are considered to by private,

- which kind of activities or data are considered to by public,

- nature of attackers observations (bisimulation, . . . , traces, . . .),

- other capabilities of attackers (time, statistic distributions, prebelief, . . .),

- qualitative vs. quantitative properties.

- state-based vs. language-based properties

What to do if a system is shown to be insecure?

Solutions

1. system redesign

Solutions

1. system redesign
2. quantification of secure

Solutions

1. system redesign
2. quantification of secure
3. supervisory control

Solutions

1. system redesign
2. quantification of secure
3. supervisory control
4. insertion functions

The aim of the paper

We propose and investigate time insertion functions which guarantee system security with respect to process opacity and timing attacks.

Working formalism: timed process algebra (TPA), which is a variant of CCS.

Working formalism: timed process algebra (TPA), which is a variant of CCS.

To add time to CCS calculus we introduce action t(-ick).

Working formalism: timed process algebra (TPA), which is a variant of CCS.

To add time to CCS calculus we introduce action t(-ick).

The resulting set of actions will be denoted as $Actt$.

Working formalism: timed process algebra (TPA), which is a variant of CCS.

To add time to CCS calculus we introduce action t(-ick).

The resulting set of actions will be denoted as $Actt$.

Operational semantics - labelled transition systems.

$$\frac{}{Nil \xrightarrow{t} Nil} \; A1 \qquad \frac{}{u.P \xrightarrow{t} u.P} \; A2$$

$$\frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q', P \mid Q \xrightarrow{\tau} \not\rightarrow}{P \mid Q \xrightarrow{t} P' \mid Q'} \; Pa \qquad \frac{P \xrightarrow{t} P', Q \xrightarrow{t} Q'}{P + Q \xrightarrow{t} P' + Q'} \; S$$

Opacity

1. What an attacker can see:

Opacity

1. What an attacker can see:

Definition (**Observation function**)

Let Θ be a set of elements called observables. Any function $\mathcal{O} : Actt^{\star} \rightarrow \Theta^{\star}$ is an observation function.

Opacity

We call an observation function as $(w = x_1 \dots x_n)$:

- static if there is a mapping $\mathcal{O}' : Actt \to \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1) \dots \mathcal{O}'(x_n)$,
- dynamic if there is a mapping $\mathcal{O}' : Actt^\star \to \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1).\mathcal{O}'(x_1.x_2) \dots \mathcal{O}'(x_1 \dots x_n)$,
- orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \to \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w).\mathcal{O}'(x_2, w) \dots \mathcal{O}'(x_n, w)$,
- m-orwellian if there is a mapping $\mathcal{O}' : Actt \times Actt^\star \to \Theta \cup \{\epsilon\}$ such that for every $w \in Actt^\star$ it holds $\mathcal{O}(w) = \mathcal{O}'(x_1, w_1).\mathcal{O}'(x_2, w_2) \dots \mathcal{O}'(x_n, w_n)$ where $w_i = x_{max\{1,i-m+1\}}.x_{max\{1,i-m+1\}+1} \cdots x_{min\{n,i+m-1\}}$.

Opacity

2. What is a private property to be deduced?

Opacity

2. What is a private property to be deduced?

Any predicate ϕ over processes.

Opacity

3. What means an absence of information flow?

Opacity

3. What means an absence of information flow?

Definition (**Process Opacity**)

Given process P, a predicate ϕ over processes is process opaque w.r.t. the observation function $\mathcal{O}$ whenever $P \xrightarrow{w} P'$ for $w \in Actt^*$ and $\phi(P')$ holds then there exists P'' such that $P \xrightarrow{w'} P''$ for some $w' \in Actt^*$ and $\neg\phi(P'')$ holds and moreover $\mathcal{O}(w) = \mathcal{O}(w')$.

The set of processes for which the predicate ϕ is opaque with respect to $\mathcal{O}$ will be denoted by $POp^{\phi}_{\mathcal{O}}$.

Timing Attacks

Function $\mathcal{O}_t$ is untimed variant of $\mathcal{O}$ iff $\mathcal{O}(w) = \mathcal{O}_t(w|_{Act})$, i.e. untimed variant represents an observer who does not see elapsing of time since both traces, with and without actions t, are seen equally.

Definition (**Timinig Attacks**)

We say that process P is prone to timing attacks with respect to ϕ and $\mathcal{O}$ iff $P \notin POp^{\phi}_{\mathcal{O}}$ but $P \in POp^{\phi}_{\mathcal{O}_t}$.

To protect systems against timing attacks we propose application of time inserting functions.

To protect systems against timing attacks we propose application of time inserting functions.

Such functions can add some idling between actions to enforce process's security.

Definition (**Time Insertion function**)

Any function $\mathcal{F} : Actt^\star \to Actt^\star$ is an insertion function iff for every $w \in Actt^*$ we have $w \ll_{\{t\}} \mathcal{F}(w)$. It is called static /dynamic /orwellian / m-orwellian ($m \geq 1$) if the following conditions hold respectively (below we assume $w = x_1 \ldots x_n$):

- static if there is a mapping $f : Actt \to \{t\}^*$ such that for every $w \in Actt^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1).x_2.f(x_2) \ldots x_n.f(x_n)$,
- dynamic if there is a mapping $f : Actt^\star \to \{t\}^*$ such that for every $w \in Act^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1).x_2.f(x_1.x_2) \ldots x_n.f(x_1.\ldots.x_n)$,
- orwellian if there is a mapping $f' : Actt \times Actt^\star \to \{t\}^*$ such that for every $w \in Actt^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1, w).x_2.f(x_2, w) \ldots x_n.f(x_n, w)$,
- m-orwellian if there is a mapping $f' : Actt \times Actt^\star \to \{t\}^*$ such that for every $w \in Actt^\star$ it holds $\mathcal{F}(w) = x_1.f(x_1, w_1).x_2.f(x_2, w_2) \ldots x_n.f(x_n, w_n)$,

Imunity

Definition

We say that process P can be immunized for process opacity with respect to a predicate ϕ over $Actt^{\star}$ and the observation function $\mathcal{O}$ if for every P', $P \xrightarrow{w} P'$ such that $\phi(P')$ holds and there does not exists P'' such that $P \xrightarrow{w'} P''$ for some w' such that $\mathcal{O}(w) = \mathcal{O}(w')$ and $\phi(P'')$ does not hold, there exist w_t, $w \ll_{\{t\}} w_t$ such that $P \xrightarrow{w_t} P''$ and and there exists P''' and w'', such that $P \xrightarrow{w''} P'''$ such that $\neg\phi(P''')$ holds and $\mathcal{O}(w_t) = \mathcal{O}(w'')$.

Results

Definition

We say that observational function $\mathcal{O}$ is not sensitive to τ action iff $\mathcal{O}(w) = \mathcal{O}(w|_{At})$ for every $w \in Act^*$. Otherwise we say that $\mathcal{O}$ is sensitive to τ action.

Results

Definition

We say that observational function $\mathcal{O}$ is not sensitive to τ action iff $\mathcal{O}(w) = \mathcal{O}(w|_{At})$ for every $w \in Act^*$. Otherwise we say that $\mathcal{O}$ is sensitive to τ action.

Theorem

Let P is prone to timing attack with respect to $\mathcal{O}$ and ϕ. Let $\tau \notin L(P)$, P is sequential (i.e. does not contain parallel composition) and $\mathcal{O}$ is static. Then P can be immunized.

Results

Definition

We say that observational function $\mathcal{O}$ is time non-contextual if $\mathcal{O}_t(w) = \mathcal{O}_t(w')$ for every w, w' such that $w \ll_{\{t\}} w'$.

Results

Definition

We say that observational function $\mathcal{O}$ is time non-contextual if $\mathcal{O}_t(w) = \mathcal{O}_t(w')$ for every w, w' such that $w \ll_{\{t\}} w'$.

Theorem

Let process P is prone to timing attacks with respect to ϕ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks.

Results

Corollary 1. Let $\mathcal{O}$ is a static observation function such that $\mathcal{O}(\tau) = \epsilon$ and $\mathcal{O}(t) = t$. Then process P which is prone to timing attacks with respect to ϕ and observation function $\mathcal{O}$ can be immunized for process opacity with respect to timing attacks.

Results

Corollary 1. Let $\mathcal{O}$ is a static observation function such that $\mathcal{O}(\tau) = \epsilon$ and $\mathcal{O}(t) = t$. Then process P which is prone to timing attacks with respect to ϕ and observation function $\mathcal{O}$ can be immunized for process opacity with respect to timing attacks.

Theorem

Let process P is prone to timing attacks with respect to ϕ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks by a m-orwellian insertion function, moreover such one, which can be emulated by finite state process.

Results

Definition

We say that predicate ϕ is time sensitive iff whenever $\phi(P)$ holds for P then there exists $n, n > 0$ such that $P \xrightarrow{t^n} P'$ and $\phi(P')$ does not hold.

Results

Definition

We say that predicate ϕ is time sensitive iff whenever $\phi(P)$ holds for P then there exists $n, n > 0$ such that $P \xrightarrow{t^n} P'$ and $\phi(P')$ does not hold.

Theorem

Let process P is prone to timing attacks with respect to time sensitive predicate $\neg\phi$ and time non-contextual observation function $\mathcal{O}$ which does not see τ. Then P can be immunized for opacity with respect to timing attacks, $\mathcal{O}$ and ϕ.

Results

Theorem

Immunizability is undecidable i.e. it cannot be decided whether P can be immunized for opacity with respect to timing attacks.

Results

Theorem

Immunizability is undecidable i.e. it cannot be decided whether P can be immunized for opacity with respect to timing attacks.

Theorem

Immunizability is decidable for static and m-orwellian observation function $\mathcal{O}$.

Conclusions

We have investigated time insertion functions for timed process algebra which enforce the security with respect to timing attacks.

Conclusions

We have investigated time insertion functions for timed process algebra which enforce the security with respect to timing attacks.

The presented approach allows us to exploit also process algebras enriched by operators expressing other "parameters" (space, distribution, networking architecture, power consumption and so on).

Thank you for your attention!