



Declassification Policy Management in Dynamic Information Systems



ARES 2011 - Vienna, Austria

Julien A. Thomas, Nora Cuppens-Boulahia and Frédéric Cuppens

Institut Télécom ; Télécom Bretagne ; SFIS Team
Université Européenne de Bretagne

22-26 August 2011

The study is partially funded by the Délégation Générale
pour l'Armement under a DGA / CNRS grant

Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
- 4 Security Policy for Dynamic I.S. with Declassification
- 5 Conclusion

Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
- 4 Security Policy for Dynamic I.S. with Declassification
- 5 Conclusion

Multilevel Information Systems

- ⊕ An information flow is the transfer of information from a variable x to a variable y in a given *process*
- ⊕ Multilevel Information Systems
 - ⊕ Process information with different sensitivities
 - ⊕ security level lattices
 - ⊕ Permit simultaneous access by users with different security clearances
 - ⊕ Prevent illegal flow from higher level to lower level

Multilevel Information Systems

- ⊕ An information flow is the transfer of information from a variable x to a variable y in a given *process*
- ⊕ Multilevel Information Systems
 - ⊕ Process information with different sensitivities
 - ⊕ security level lattices
 - ⊕ Permit simultaneous access by users with different security clearances
 - ⊕ Prevent illegal flow from higher level to lower level
- ⊕ Security properties: specify the authorized flows
- ⊕ Security policy: enforce the security properties

Dynamic Information Systems

- ⊕ Dynamic Mechanisms
 - ⊕ Databases: Triggers
 - ⊕ Event based information systems
 - ⊕ An action $\alpha_1(P)$ triggers another action $\alpha_2(P_2)$, providing ...

Dynamic Information Systems

- ⊕ Dynamic Mechanisms
 - ⊕ Databases: Triggers
 - ⊕ Event based information systems
 - ⊕ An action $\alpha_1(P)$ triggers another action $\alpha_2(P_2)$, providing ...
- ⊕ Model for Dynamic Information Systems:
 - Event-Condition-Action-rules* and the L_{active} language

- ⊕ actions

```
do( $\alpha$ ) causes op1, ..., opk if q1(X1), ... qn(Xn);
```

- ⊕ events

```
event_name(X) after do( $\alpha$ ) if r1(X1) rm(Xm) ;
```

- ⊕ rules

```
rule: event_name(X) initiates do( $\alpha_1$ ), ... do( $\alpha_k$ )
if t1(X1), ... tp(Xp);
```

Dynamic Multilevel Information Systems

- ⊕ Multilevel ECA rules [TCBC10]
 - ⊕ Multilevel model
 - ⊕ for confidentiality **without declassification**

ECA Rules Security

$do(\alpha, L)$ causes $f(Y)$ if $q_1(X_1) \dots q_n(X_n)$
 $event_name(X, L1)$ after $do(\alpha, L2)$ if $r_1(Z_1) \dots r_m(Z_m)$
 $rule_name : event_name(X, L_0)$ initiates $do(\alpha_1, L_1) \dots do(\alpha_n, L_n)$
 if $t_1(Z_1) \dots t_m(Z_m)$

- ⊕ Confidentiality property and policy

Dynamic Multilevel Information Systems with Declassifications

- ⊕ Insecure information flows in dynamic information systems
 - ⊕ Consider the following declassification policy:
 - ⊕ $warCost(C)$ is a TS predicate that specifies the cost of a war,
 - ⊕ $number_death$ is a S predicate that will never be downgraded,
 - ⊕ $declassify_event(warCost(Cost), \perp)$ after $do(insert(victory))$ if $endOfWar$

Dynamic Multilevel Information Systems with Declassifications

- ⊕ Insecure information flows in dynamic information systems
 - ⊕ Consider the following declassification policy:
 - ⊕ $warCost(C)$ is a TS predicate that specifies the cost of a war,
 - ⊕ $number_death$ is a S predicate that will never be downgraded,
 - ⊕ $declassify_event(warCost(Cost), \perp)$ after $do(insert(victory))$ if $endOfWar$
 - ① $victory, endOfWar$: downgrading depends on contextual events whose occurrence triggers declassification
Integrity of victory must be guaranteed on declassification
 - ② declassification may be responsible for the indirect insecure declassifications: concrete value of $warCost(C)$?
Integrity of $warCost(C)$ must be guaranteed on declassification

Dynamic Multilevel Information Systems with Declassifications

- ⊕ Insecure information flows in dynamic information systems
 - ⊕ Consider the following declassification policy:
 - ⊕ $warCost(C)$ is a TS predicate that specifies the cost of a war,
 - ⊕ $number_death$ is a S predicate that will never be downgraded,
 - ⊕ $declassify_event(warCost(Cost), \perp)$ after $do(insert(victory))$ if $endOfWar$
 - ① $victory, endOfWar$: downgrading depends on contextual events whose occurrence triggers declassification
Integrity of victory must be guaranteed on declassification
 - ② declassification may be responsible for the indirect insecure declassifications: concrete value of $warCost(C)$?
Integrity of $warCost(C)$ must be guaranteed on declassification
- ⊕ Specifying when a declassification flow is secure

Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
- 4 Security Policy for Dynamic I.S. with Declassification
- 5 Conclusion

Declassification Model

- ⊕ Based on ECA rules specification

Declassification

```
do(declassify(P,L)) causes  
delete(classification_level(P,L')), insert(classification_level(P,L))  
if classification_level(P,L')  $\wedge$  inf_level(L,L');
```

- ⊕ Previous work of Thomas and al. [TCBC09]
 - ⊕ Trust Based Downgrading: A user u is allowed to downgrade a piece of information o if u is trusted for the downgrading order on o .

Trust based downgrading

- ⊕ As an access control requirement
- ⊕ $is_permitted \in USERS \times ACTIONS$

Declassification Model

- ⊕ Based on ECA rules specification

Declassification

```
do(declassify(P,L)) causes
delete(classification_level(P,L')), insert(classification_level(P,L))
if classification_level(P,L')  $\wedge$  inf_level(L,L');
```

- ⊕ Previous work of Thomas and al. [TCBC09]
 - ⊕ Trust Based Downgrading:
 - ⊕ Automatic Downgrading: the downgrading operation is based on the occurrence of a specific event

Automatic downgrading

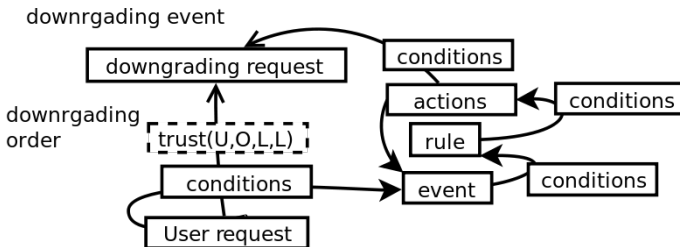
- ⊕ `declassify_rule: declassify_event(P,L) initiates do(declassify(P,L));`
- ⊕ Specifying when `declassify_event(P,L)` occurs
- ⊕ `declassify_event(warCost(Cost), \perp) after do(insert(victory)) if endOfWar`

Declassification Model

- Based on ECA rules specification

Declassification

```
do(declassify(P,L)) causes
delete(classification_level(P,L')), insert(classification_level(P,L))
if classification_level(P,L')  $\wedge$  inf_level(L,L');
```



Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
 - Formal Model for Security Properties
 - Security Properties
- 4 Security Policy for Dynamic I.S. with Declassification
- 5 Conclusion

Trace model for security properties

⊕ Trace model

⊕ A trace $t = \{s_0, \alpha\}$ is composed of

⊕ the initial state s_0

⊕ an infinite sequence α of actions executed in trace t .

$t_n = \{s_0, \alpha_n\}$ is a finite trace composed of the initial state s_0 and the sequence α_n of the n first actions executed in t .

⊕ If $t_n = \{s_0, \alpha_n\}$ is a finite trace, then s_n is the obtained state

Trace model for security properties

- ⊕ Trace model for security [BC91]
 - ⊕ Each action in a trace is executed at a given security level
 - ⊕ $\alpha \upharpoonright L$: subsequence of actions whose security level is lower than or equal to L .
 - ⊕ Two states s and s' are equivalent at level L : $s \approx_L s'$
 - ⊕ The classification of predicates classified lower than or equal to L is identical in s and s'
 - ⊕ Their truth value is identical in s and s' .

Trace model for security properties

- ⊕ Trace model for security [BC91]
 - ⊕ Each action in a trace is executed at a given security level
 - ⊕ $\alpha \upharpoonright L$: subsequence of actions whose security level is lower than or equal to L .
 - ⊕ Two states s and s' are equivalent at level L : $s \approx_L s'$
 - ⊕ The classification of predicates classified lower than or equal to L is identical in s and s'
 - ⊕ Their truth value is identical in s and s' .
- ⊕ Trace equivalence: users' privileges

$$t \sim_L t' \Leftrightarrow s_0 \approx_L s'_0 \wedge \alpha \upharpoonright L = \alpha' \upharpoonright L.$$

Security Properties

- ⊕ Without declassification actions: users privileges and actions
 - ⊕ $ext(t)$: external (users) actions of t
 - ⊕ $ext(t) \sim_L ext(t') \Leftrightarrow s_0 \approx_L s'_0 \wedge ext(\alpha)[L = ext(\alpha')][L$

Security Properties

- ⊕ Without declassification actions: users privileges and actions
 - ⊕ $ext(t)$: external (users) actions of t
 - ⊕ $ext(t) \sim_L ext(t') \Leftrightarrow s_0 \approx_L s'_0 \wedge ext(\alpha)[L = ext(\alpha')][L$
 - ⊕ Confidentiality property [TCBC10]

$$1 \quad ext(t_n) \sim_L ext(t'_{n'}) \implies \alpha_n [L = \alpha'_{n'}][L$$

$$2 \quad t_n \sim_L t'_{n'} \implies s_n \approx_L s'_{n'}.$$

- ⊕ formalize other proposals: Bell and Lapadula, Non Interference

Security Properties

⊕ Without declassification actions: users privileges and actions

⊕ $ext(t)$: external (users) actions of t

⊕ $ext(t) \sim_L ext(t') \Leftrightarrow s_0 \approx_L s'_0 \wedge ext(\alpha)[L = ext(\alpha')][L$

⊕ Confidentiality property [TCBC10]

$$1 \quad ext(t_n) \sim_L ext(t'_{n'}) \implies \alpha_n[L = \alpha'_{n'}][L$$

$$2 \quad t_n \sim_L t'_{n'} \implies s_n \approx_L s'_{n'}$$

⊕ formalize other proposals: Bell and Lapadula, Non Interference

⊕ Integrity property

$$\oplus \quad t \sim_L t' \Leftrightarrow s_0 \approx_L s'_0 \wedge \alpha[L = \alpha'] [L$$

$$\oplus \quad t_n \sim_L t'_{n'} \implies s_n \approx_L s'_{n'}$$

Security Properties

⊕ With declassification actions

- ⊕ $t = (s_o, \alpha)$ and $t' = (s_o, \alpha')$ are declassification equivalent (\approx_{L_D}) if
 - 1 The subsequence of declassification actions at L is identical in t and t'
 - 2 For every declassification action $d = \text{declassify}(P, L)$, if $\text{indexof}(d, \alpha) = i$ and $\text{indexof}(d, \alpha') = i'$ then $\Pi(P, s_i) = \Pi(P, s_{i'})$

Security Properties

⊕ With declassification actions

- ⊕ $t = (s_0, \alpha)$ and $t' = (s_0, \alpha')$ are declassification equivalent (\approx_{L_D}) if
 - 1 The subsequence of declassification actions at L is identical in t and t'
 - 2 For every declassification action $d = \text{declassify}(P, L)$, if $\text{indexof}(d, \alpha) = i$ and $\text{indexof}(d, \alpha') = i'$ then $\Pi(P, s_i) = \Pi(P, s_{i'})$

$$t \sim_{L_D} t' \Leftrightarrow s_0 \approx_L s'_0 \wedge \alpha \upharpoonright_{L_D} = \alpha' \upharpoonright_{L_D} \wedge t \approx_{L_D} t'$$

Security Properties

⊕ With declassification actions

- ⊕ $t = (s_o, \alpha)$ and $t' = (s_o, \alpha')$ are declassification equivalent (\approx_{L_D}) if
 - 1 The subsequence of declassification actions at L is identical in t and t'
 - 2 For every declassification action $d = \text{declassify}(P, L)$, if $\text{indexof}(d, \alpha) = i$ and $\text{indexof}(d, \alpha') = i'$ then $\Pi(P, s_i) = \Pi(P, s'_{i'})$

$$t \sim_{L_D} t' \Leftrightarrow s_o \approx_L s'_o \wedge \alpha \upharpoonright L_D = \alpha' \upharpoonright L_D \wedge t \approx_{L_D} t'$$

Confidentiality in Information Systems with Declassification

- 1 $\text{ext}(t_n) \sim_{L_D} \text{ext}(t'_{n'}) \implies \alpha_n \upharpoonright L_D = \alpha'_{n'} \upharpoonright L_D$
- 2 $t_n \sim_{L_D} t'_{n'} \implies s_n \approx_L s'_{n'}$.

Integrity in Information Systems with Declassification

$$\text{ext}(t_n) \sim_L \text{ext}(t'_{n'}) \wedge t_n \sim_{L_D} t'_{n'} \implies t_n \approx_{D_L} t'_{n'}$$

Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
- 4 Security Policy for Dynamic I.S. with Declassification**
 - Integrity Policy
 - Declassification Policy for ECA rules
- 5 Conclusion

Integrity Policy

- ⊕ Control declassification by evaluating the integrity of the information
- ⊕ Enforce “standard” integrity property
 - ⊕ After an interaction, the integrity levels of the targets (triggered actions and events) are lower than the integrity levels of the sources

Condition Security - Integrity Evaluation Law

the integrity is always decreasing throughout the dependency chain.

Action Effect Security - Integrity Policy

An object *obj* may be modified by an action with an integrity level Li if its integrity level Ls satisfies $Ls \sqsubseteq Li$.

- ⊕ Sufficient conditions to enforce integrity for dynamic multilevel information systems without declassification

Integrity Policy for ECA rules

Action Integrity Security

the execution of an action at the integrity level Li satisfies both the Integrity Evaluation law and the Integrity policy

Event Integrity Security

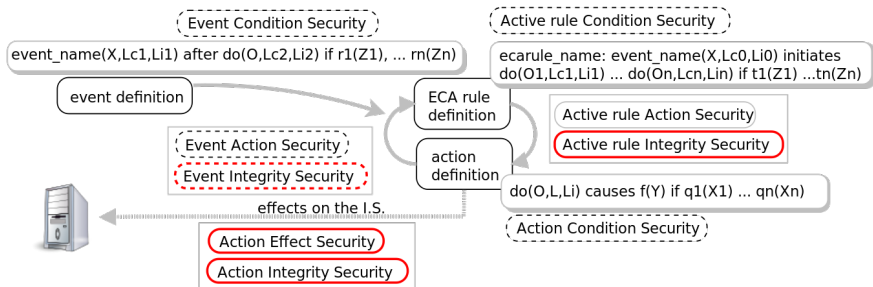
the integrity level Li_e of the event is lower than or equal to the lower bound of both the integrity levels of the event conditions and the integrity level Li of α .

Active Rule Integrity Security

the integrity level Li_k of each action is lower than or equal to the lower bound of the integrity levels of both the active rule conditions and the integrity level Li_e of the event.

Integrity Policy for ECA rules

- Extended Multilevel ECA model
- Additional security requirements for secure declassification ?



Declassification Policy for ECA rules

Safety Property

An object *obj* is said to be safe w.r.t. a security level L if the integrity level of *obj* being Li , $L \sqsubseteq Li$

- ⊕ The trustworthiness in a condition or an object depends on the security level of the object to declassify,

Secure Declassification Operation Conditions

When considering the declassification of P , an action, event or rule satisfies the *Secure Declassification Operation Conditions* if its confidentiality level Lc satisfies $Lc = \perp$ and its integrity level Li satisfies $classification_level(P) \sqsubseteq Li$.

Declassification Policy for ECA rules

Trust based downgrading specification

Let $do(declassify(P, L_{dest}), L_c, L_i)$ be a query executed by U .

- 1 P satisfies the safety property
- 2 $is_permitted(U, declassify(P, L_{dest}))$

Automatic downgrading specification

Let this declassification be triggered by $declassify_event(P, L_{dest}, L'_c, L'_i)$.

- 1 P satisfies the safety property
- 2 the event satisfies the Secure Declassification Operation Conditions.

- ⊕ Sufficient conditions to enforce confidentiality and integrity for dynamic multilevel information systems with declassifications

Outline

- 1 Context
- 2 Declassification Model
- 3 Security Properties for Dynamic I.S. with Declassification
- 4 Security Policy for Dynamic I.S. with Declassification
- 5 Conclusion

Conclusion

➔ Contributions

- ➔ An information flow security model based on traces to
 - ➔ manage declassifications
 - ➔ control integrity policy of information to declassify

Conclusion

⊕ Contributions

- ⊕ An information flow security model based on traces to
 - ⊕ manage declassifications
 - ⊕ control integrity policy of information to declassify
- ⊕ A security policy to enforce this model
 - ⊕ security requirements for the confidentiality and integrity of the information flows
 - ⊕ security requirements for the declassification flow
 - ⊕ proofs based on the formal B method

Conclusion

➔ Contributions

- ➔ An information flow security model based on traces to
 - ➔ manage declassifications
 - ➔ control integrity policy of information to declassify
- ➔ A security policy to enforce this model
 - ➔ security requirements for the confidentiality and integrity of the information flows
 - ➔ security requirements for the declassification flow
 - ➔ proofs based on the formal B method




➔ Implementation

- ➔ Enforcement in real environment based on weavings of the policies:
Oracle VPDs and automatic generation of triggers

➔ Future Works

- ➔ Consistency issues associated to declassification chains

References I

-  Pierre Bieber and Frédéric Cuppens, *A definition of secure dependencies using the logic of security*, CSFW, 1991, pp. 2–11.
-  Julien Thomas, Nora Cuppens-Boulahia, and Frédéric Cuppens, *Modeling and Controlling Downgrading Operations in Information Systems*, 5th International Conference on Signal-Image Technology & Internet-based Systems (SITIS'09), december 2009.
-  _____, *Expression and Enforcement of Confidentiality Policy in Active Databases*, MEDES 2010, 5th International ACM Conference on Management of Emergent Digital EcoSystems, 26 October - 29 October 2010, Bangkok, Thailand, LUSI - Dépt. Logique des Usages, Sciences Sociales et de l'Information (Institut Télécom-Télécom Bretagne), 2010.