Using Equivalence Relations for Corrective Enforcement of Security Policies

Raphaël Khoury Nadia Tawbi

Laval University

September 8, 2010

Motivation

- A monitor is a software which observes the execution of a target and reacts to prevent a violation of a security policy.
- For many applications, aborting the execution when problems are detected is not a suitable option.
- Corrective enforcement gives the monitor the ability to alter the input sequence to assure the compliance of the execution with the security policy.
- We seek an adequate model to describe such a security policy enforcement by monitors.
- This will allow us to better reason about the policies enforceable by such monitors, and the constraints imposed on their enforcement.

Outline

Outline

Framework and Review of the Literature

- A system is modeled by a (possibly infinite) set of actions Σ.
- An execution is a finite or infinite sequence of actions from Σ.
- A security policy $\hat{\mathcal{P}}$ is a subset of valid sequences.
- The monitor is an automaton which receives a sequence as input, and outputs another sequence.
- We let σ, τ and υ range over possible executions, and $\mathcal{A}(\sigma)$ denote the output of the monitor, when its input is σ. We write $\hat{\mathcal{P}}(\sigma)$ to indicate that the sequence σ respects the security policy $\hat{\mathcal{P}}$.

Framework and Review of the Literature

When can we consider that a monitor enforces a security Policy? An effective enforcement paradigm must be based on the following 2 principles (from Ligatti et al.):

Correction The output sequence is valid.

Transparency The semantics of a valid input sequence is preserved. An equivalence relation between executions limits the monitor's ability to transform sequences.

Framework and Review of the Literature

First idea: precise enforcement (from Schneider, Ligatti et al.) Every action of a valid sequence must be output in lockstep. $\forall \sigma \in \Sigma^{\infty}$

- $\hat{\mathcal{P}}(\mathcal{A}(\sigma))$
- $\hat{\mathcal{P}}(\sigma) \Rightarrow \forall i : \mathcal{A}(\sigma_i)$

Framework and Review of the Literature

Second Idea: effective$_{\cong}$ enforcement (from Ligatti et al.) The output must be equivalent if the input is valid. $\forall \sigma \in \Sigma^{\infty}$

1. $\hat{\mathcal{P}}(\mathcal{A}(\sigma))$
2. $\hat{\mathcal{P}}(\sigma) \Rightarrow \mathcal{A}(\sigma) \cong \sigma$

Outline

Corrective$_{\cong}$ Enforcement

1. The monitor should be both :
   - required to output a valid sequence, and
   - forbidden from altering the semantics of the input.
2. Valid behaviors present in an invalid input sequence should be preserved, while minimal alterations are made to correct the input sequence.
3. These valid behaviors are captured by an equivalence relation.

Corrective$_{\cong}$ Enforcement

1. Corrective$_{\cong}$ Enforcement
2. An equivalence relation captures essential properties of the input sequence, which must be preserved, despite the monitor's transformations.
3. The output must always be kept equivalent to the input.
4. $\forall \sigma \in \Sigma^{\infty}$
   - $\hat{\mathcal{P}}(\mathcal{A}(\sigma))$
   - $\sigma \cong \mathcal{A}(\sigma)$

Corrective$_{\cong}$ Enforcement

1. Constraints must be imposed on the possible equivalence relations.
   - Schneider suggests consistency:
     - $\forall \sigma, \sigma' \in \Sigma^{\infty} : \sigma \cong \sigma' \Rightarrow \hat{\mathcal{P}}(\sigma) \Leftrightarrow \hat{\mathcal{P}}(\sigma')$.
   - This is too restrictive for corrective enforcement. If two sequences are coherent, a valid sequence can never be a suitable replacement to an invalid sequence.
2. We use an abstraction function $\mathcal{F}$ to capture the property of the input sequence which must be preserved throughout the manipulations performed by the monitor.
3. We let $\leq$ stand for a partial order over the codomain of $\mathcal{F}$, and $\sqsubseteq$ stand for a corresponding partial order over the possible execution sequences, s.t. for any two sequences σ, σ'
   - $\sigma \sqsubseteq \sigma' \Leftrightarrow \mathcal{F}(\sigma) \leq \mathcal{F}(\sigma')$.
4. The abstraction, rather than the equivalence relation, must be consistent with the property.
   - $\mathcal{F}(\sigma) = \mathcal{F}(\sigma') \Rightarrow \hat{\mathcal{P}}(\sigma) \Leftrightarrow \hat{\mathcal{P}}(\sigma')$

Corrective$_{\cong}$ Enforcement

1. We define a partial order $\sqsubseteq$ over the values of $\mathcal{F}$. The equivalence relation groups together intervals of values on this partial order.
   - $\sigma \sqsubseteq \sigma' \sqsubseteq \sigma'' \wedge \sigma \cong \sigma'' \Rightarrow \sigma \cong \sigma'$
2. The greatest lower bound of two equivalent sequences is also equivalent to them.
   - $\forall \sigma, \sigma' \in \Sigma^* : \sigma \cong \sigma' \Rightarrow \exists \tau \in \Sigma^* : \tau = (\sigma \sqcap \sigma') \wedge \tau \cong \sigma$
3. $\mathcal{F}$ abstracts the behavior of interest to be preserved in a sequence, $\sqsubseteq$ organizes the sequences accordingly, $\hat{\mathcal{P}}$ establishes that only certain values of $\mathcal{F}$ are valid or that a certain threshold must be reached, while $\cong$ groups the sequences if their abstractions are equivalent.

Corrective$_{\cong}$ Enforcement

1. We define the equivalence over finite sequences. Two infinite sequences are equivalent if they have infinitely many equivalent prefixes. Let σ and σ' be infinite sequences.
   - $\forall \sigma, \sigma' \in \Sigma^{\omega} : \sigma \cong \sigma' \Leftrightarrow \forall \tau \prec \sigma : \exists \upsilon \succeq \tau : \exists \tau' \prec \sigma' : \upsilon \cong \tau'$
2. Equivalence relations must respect the following closure restriction
   - $\tau \cong \tau' \Rightarrow \tau;\sigma \cong \tau';\sigma$

Outline

Examples:Transactional Properties

Transactional Properties:

1. A valid sequence is composed of finite or infinite repetition of finite factors from a set of valid transactions.
2. These properties model the behavior of iterative systems such as ATMs or online stores.
3. Prior research suggested to enforce this property by aborting the execution when an invalid transaction is encountered, or using ad hoc enforcement solutions applicable to this property only, and to a specific enforcement mechanism only.

Examples:Transactional Properties

1. Corrective$_{\simeq}$ enforcement allows the monitor to output valid transactions while deleting invalid ones.
   - The abstraction is the multiset of transactions present in the sequence.
   - The partial order is multiset inclusion of transactions.
   - Two sequences are equivalent if they share the same multiset of valid transactions.
2. To be enforceable in this manner, a transactional property must meet the following property, termed unambiguity.
   - $\forall \sigma, \sigma' \in \mathcal{T} : \forall \tau \in \mathit{pref}(\sigma) : \forall \tau' \in \mathit{suf}(\sigma') : \tau \neq \epsilon \wedge \tau' \neq \epsilon \Rightarrow \tau;\tau' \notin \mathcal{T}$
3. We prove that this condition is both necessary and sufficient for the enforcement of a transactional property.

Examples: Renewal Properties

1. The set of Renewal properties includes all properties for which infinite valid sequences include infinitely many valid prefixes, while infinite valid sequences include only finitely many such prefixes.
2. This set corresponds to 5 of the 6 classes of the safety-progress hierarchy of properties.
3. This is the set of properties which can be effectively$_{=}$ enforced.

Example: Renewal Properties

1. Any Infinite Renewal Property can be enforced in this manner.
   - The abstraction function is the identity function.
   - The partial order is the prefix comparison.
   - Two sequences are equivalent if they share the same longest valid prefix, w.r.t. the property of interest.
2. This allows us to characterize effective enforcement as a special case of corrective$_{\simeq}$ enforcement, with a particular equivalence relation.

Outline

Nonuniform Enforcement

1. Prior research has established that the set of properties enforceable by monitors can be extended by drawing upon a static analysis of the target program.
   - The use of an a priori static analysis can extend the set of properties which are correctively$_{\cong}$ enforceable.
   - This does not occur monotonously, and in some cases, a static analysis provides no benefits.
   - There is a monotonous increase in the set of enforceable properties iff the equivalence relation is syntactic equality.

Outline

Conclusion

1. We propose a new framework to model the behavior of a corrective monitor.
2. We use equivalence relations to restrict the monitors ability to transform its input so than the input's semantics is preserved.
3. The result is a model of a monitor which can correct invalid sequences but which preserves valid behavior already present in its input.
4. We proved several theorems relating to the set of properties enforceable by such monitors under various constraints, and using different equivalence relations.
5. Future work should focus on developing automatic methods to inline a corrective monitor inside a program. We also aim at studying the possibility of replacing equivalence relations with partial orders. This will allow the monitor to output an approximation of the input sequence. Work in this direction will be presented at the FAST 2010 Workshop.

Thank You

Questions?