Predictability of Enforcement*

Nataliia Bielova and Fabio Massacci

University of Trento, Italy, lastname@disi.unitn.it

Abstract. The current theory of runtime enforcement is based on two properties for evaluating an enforcement mechanism: *soundness* and *transparency*. Soundness defines that the output is always good ("no bad traces slip out") and transparency defines that good input is not changed ("no surprises on good traces"). However, in practical applications it is also important to specify how bad traces are fixed so that the system exhibits a reasonable behavior. We propose a new notion of *predictability* which can be defined in the same spirit of continuity in real-functions calculus. It defines that there are "no surprises on bad input". We discuss this idea based on the feedback of an industrial case study on e-Health.

1 Introduction

Run-time monitoring is a well known technique to control an untrusted application that runs in an otherwise secure environment. During its run the application receives some inputs from the environment and produces some (tentative) outputs. If the monitor considers these i/o sequences legal according to some policy then it will let them pass, otherwise it will block them or somehow change them. This simple and intuitive description applies for a variety of monitors: from usage control and DRM [16], to control of downloaded applications in .NET mobile code [7], or from Javascript confinements in browsers [17] to Enterprise Service Bus enforcement for web services [10]. It can be implemented by wrappers as in Google's Caja, by inlining hooks with a separate policy decision point as in [7] or by inlining the monitor code as in Polymer or PSLANG [8].

While many research proposals and tools exist on the market, we have not found major deployments in the field. A reason may be that the overhead penalty is still significant but, based on our experience on a concrete case study, we argue that there is a deeper, more foundational reason.

Italian hospitals must guarantee the compliance of many business processes involving large amounts of money (reimbursements from public health authorities for drugs dispensation), significant privacy concerns (drugs can be related to HIV or other serious illnesses), major safety considerations (drugs might have serious side-effects), and compliance with many regulations. These regulations

* We would like to thank Marta Zambetti, Marco Nalin, Andrea Micheletti and Daniela Marino from the Hospital San Raffaele for many useful discussions that helped to shape our proposal. This work has been partly supported by the EU under the projects EU-IP-MASTER, EU-FET-IP-SecureChange and EU-NoE-NESSoS.

can change frequently and a run-time monitor could guarantee the compliance of each process with minor efforts: no change to processes or ESB architectures, we deploy an updated policy and we are done.

Unfortunately, a monitor must offer some guarantees to the hospital on what happens when things are not according to the policy. This is the point where we found a foundational gap. At present the only offered formal guarantees are

transparency the monitor will never touch a good execution;
soundness the monitor will never output a trace violating the policy.

These properties are currently used in all state-of-the-art papers on runtime enforcement [4, 9, 14, 20, 21]. The recently introduced property of completeness [14] is implied by transparency and soundness. In all practical settings these two properties are necessary but not sufficient.

Reality Check 1 *What the risk manager of the hospital wants to know is "What the monitor* normally *does when a doctor's action does not respect the policy?" Does it abort the whole transactions if a research protocol number is not entered (Schneider Security Automata [19])? Does it alert the head of the department if we are prescribing a drug out of stock (Pretschner's usage control [18])? Does it wait till the doctor opens the therapeutical notes before committing the transaction to the audit logs (Ligatti's longest valid prefix automata [13])?*

While we can implement a concrete enforcement monitor to give the desired answer, there is no general, principled guarantee in the same way that we have for soundness and transparency.

Reality Check 2 *In medical terms, the "protocol" in charge of caring for bad traces is too underspecified (the nurse will somehow deliver* a *right drug). Tolerance of error is accepted in the medical domain. Often drugs cannot be dosed exactly at the milliliter, yet one must be able to give some indication of the errors (or safety margins) that the protocol might tolerate. Here the only thing we could say that it depends on the nurse dispensating the drug (i.e. our particular implementation of the run-time monitor).*

1.1 The contribution of this paper

To address this problem we propose a new theoretical notion beside transparency and soundness to describe the behavior of the run-time monitor when it takes care of bad traces. To this extent we need two contributions:

- A formal notion of distance to clarify the meaning of "being close" to the original input or to a legal trace
- The equivalent formal $\epsilon - \delta$ notion of boundedness and continuity that we have in the real Calculus.

Once we have a notion of distance we can generalize the notions of transparency and soundness to weaker notions that apply to bad and good traces

Table 1. Properties of enforcement mechanisms.

Name	**Brief description**
Soundness	every trace is mapped into *some* valid trace
Transparency	every valid trace is mapped into itself
Bounded Map	every trace is mapped into a trace close to *one* valid trace
Boundedness	every trace is mapped into a trace close to *some* valid trace
Conditional Boundedness	every trace that is close to some valid trace is mapped into a trace close to *some* (possible other) valid trace
Predictability	every trace that is close to some valid trace is mapped into a trace close to *the same* valid trace

alike: bounded map, boundedness, conditional boundedness and (our final proposal) predictability. We give their brief descriptions in Table 1.

Boundedness corresponds intuitively to a weakening of soundness: every trace should be mapped not to some valid trace but to an "almost valid" trace, which is close to the valid one. Transparency is only defined for valid traces (they should not be changed) and by weakening it we define conditional boundedness, which specifies that "almost valid" traces should be mapped to (possibly other) "almost valid" traces. Both these notions are not sufficient to deal with real life situations. Predictability is the notion that deals with traces close to valid ones and maps them to the predictable (closest) output.

The next section presents our running example (§2). Then we introduce some standard notation (§3) and discuss the applicability of various metrics(§4). The next steps (§5-6) present and discuss the different notions generalizing soundness and transparency from Table 1. We conclude the paper by discussing some open problems (§7).

2 Running example

The case study is based on a healthcare process of drug dispensation. Hospitals accredited with the Public National Health Service are in charge of administering drugs and providing diagnostic services to patients. These hospitals are obliged to claim the cost of drug dispensation or diagnostic services to the Regional Healthcare Authority.

In the region of Lombardia, the process called "File F" is used by hospitals to refund the drugs administered and/or supplied by the hospitals' outpatient departments to the patients that are not hospitalized (we sketch some steps of the process in Fig. 1). This process is highly regulated and requirements are often subject to changes. Regulations span from national or regional health service legislation related to drug reimbursements to data protection laws, from health specific standards such as HL7 to ISO-type standards or whose adoption follows from compliance to best practices.

In order to give an idea of the sheer volume of regulation, the simple process of authorization and accounting for the dispensation and reimbursement of drugs is subject to the following (not exhaustive) set of (local) regulations interpreting

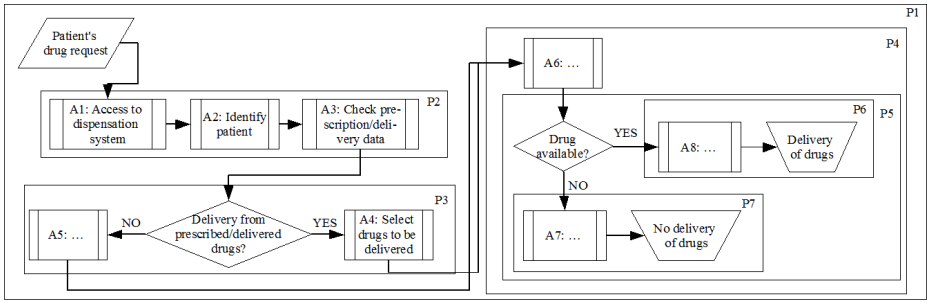

Fig. 1. The Top-Level process

national and EU laws: regional directive 17/SAN 3.4.1997, amended by directive No. 5/SAN 30_1_2004, Circular No. 45/SAN 23_12_2004, Note 30.11.2007 H1.2007.0050480, Note 27.3.2008 H1.2008.0012810, and Note 04.12.2008 H1.-2008.0044229 etc. Already in this partial sample we have had a quick turn-around of less than 6 months. In many cases compliance has to be almost immediate. For this kind of processes run-time monitors could be an effective solution.

Our running example is the drug selection subprocess of the drug dispensation process (denoted with A4 in Fig. 1). Its main steps are the following ones (in brackets we write the abbreviation of the step used in the paper):

1. The doctor fills list of drugs for his patient and selects one drug (Dis).
2. If the drug is highly sensitive, reviewing therapeutical notes is needed. They will be shown to the doctor and he has to review them (Tnn; Rtn). Otherwise therapeutical notes can be emitted (TnNn).
3. The system checks drug's submission to Research program and in case the drug is registered (Dr) shows the notification to the doctor. Then the doctor should insert the research protocol number (Irpn), a number of the protocol according to which the drug can be given to the patient. If the drug is not registered to Research program, then the doctor skips this step (DNr).
4. The doctor performs other actions needed for the drug prescription (Dpres).

3 Standard notations of enforcement

The set of observable process actions is denoted by Σ and a set of possible actions to be executed is T. A *trace* is a finite or infinite sequence of actions; the set of all finite sequences over Σ is denoted by Σ^*, the set of infinite sequences is Σ^ω, and the set of all sequences is Σ^∞. By σ we refer to a trace and by $\cdot$ we refer to an empty trace. We write $\sigma;\tau$ to denote concatenation of two sequences, where σ must be finite. A trace consisting of actions requested for execution is a *tentative execution*. A runtime monitor $E : \Sigma^\infty \to \mathrm{T}^\infty$ transforms tentative executions into a sequences of actions that will be actually executed on the system.

A *security policy* is a set of traces $P \subseteq \Sigma^\infty$. A policy P is a *security property* if there exists a predicate $\widehat{P}$ over the traces, such that $\forall\sigma \in \Sigma^\infty : \widehat{P}(\sigma) \Leftrightarrow \sigma \in P$.

Reality Check 3 *Information-flow policies cannot be evaluated by looking at a single trace but must be evaluated by comparing a trace with other possible executions. This characteristic makes them totally "un-interesting" for stakeholders. They (or their hospital) can be held financially or penally liable only for what actually happened, so only actual traces matter.*

So in the rest of the paper we use interchangeably the policy P with its corresponding predicate $\widehat{P}$. The trace σ that satisfies the property $\widehat{P}$ is called *valid*, and the trace that does not satisfy the property is called *invalid*.

Example 1. The security policy P consists of the following traces:

SimpleRun Dis; TnNn; DNr; Dpres,
NoteRun Dis; Tnn; Rtn; DNr; Dpres,
ResearchRun Dis; TnNn; Dr; Irpn; Dpres,
NoteResearchRun Dis; Tnn; Rtn; Dr; Irpn; Dpres,

and their closure under concatenation: for every $\sigma, \sigma' \in P : \sigma;\sigma' \in P$. Notice that an empty trace **NoRun** also satisfies the policy.

Example 2. Let us now make some examples of invalid traces with respect to the policy P. The doctors might forgot to click the "I have read the Therapeutical Note" button and rather close the window (Ctw). A similar event could happen for the step in which research protocol numbers are not inserted (Cpw), or he might skip all steps altogether. These alternatives give us the following traces

CloseProt Dis; TnNn; Dr; Cpw; Dpres
SkipAll Dis; Dr; Dpres
CloseNoteProt Dis; Tnn; Ctw; Dr; Cpw; Dpres

Definition 1. *An enforcement mechanism E is* sound *iff* $\forall \sigma \in \Sigma^\infty : E(\sigma) \in P$. *It is* transparent *iff* $\forall \sigma \in \Sigma^\infty : (\sigma \in P \Rightarrow E(\sigma) = \sigma)$.

Fig. 2 graphically shows soundness and transparency, inputs on the left side of the figure marked with Σ^∞ and outputs on the right side marked with T^∞. The gray area denotes invalid sequences and the white area denotes valid ones.

Transparency means that the traces in the white area of the input are mapped into *the same* traces (at the same position) in the white area of the output. Soundness means that all the traces shall be mapped into the white area. However, it is not specified where exactly the points from gray area are mapped.

The valid traces ResearchRun and NoteResearchRun are mapped into the same traces in the output, while invalid ones (CloseProt, SkipAll, CloseNoteProt) are mapped into good traces that are chosen arbitrarily.

In the original definition of Bauer et al. [2] the equal ("≈") relation is used in the right part of the last statement. In another recent work Khoury and Tawbi [11] discussed possible semantics of this relation, however it is not possible to define a semantics that can be used in all domains. Hence, we use an equivalence relation in order to compare this definition with the new notions we propose in this paper. Ligatti and Reddy [14] have proposed the notion of completeness instead. It can be easily shown that transparency implies completeness and since transparency is necessary here, we don't discuss it further.

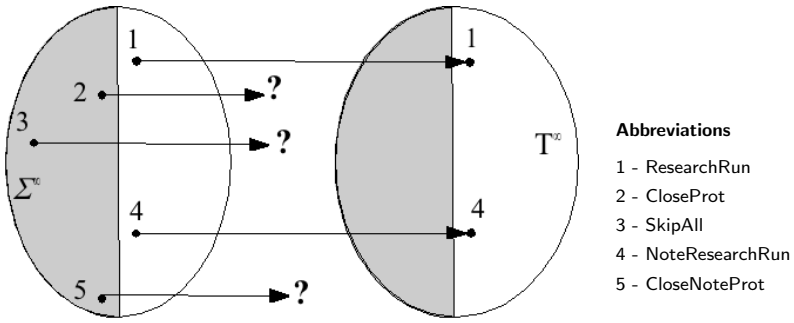

Fig. 2. Sound and Transparent enforcement mechanism.

4 Metrics and distances

Definition 2. *A* metric[1] *on a set S is a function $d : S \times S \to \mathbb{R} \cup \{\infty\}$ such that (a) $d(\sigma, \tau) \geq 0$, (b) $d(\sigma, \tau) = 0$ if and only if $\sigma = \tau$, (c) symmetry: $d(\sigma, \tau) = d(\tau, \sigma)$, (d) triangular inequality: $d(\sigma, \tau) \leq d(\sigma, \sigma') + d(\sigma', \tau)$.*

The pair (S, d) is called a *metric space* and the number $d(\sigma, \tau)$ is called the *distance between the elements* σ and τ. If all conditions but symmetry hold, then d is called a *quasi-metric*. We propose several concrete metrics that will be used in the paper. Each of them has passed our reality check.

Reality Check 4 *When a distance is a finite number we can compare how far two situations (some potentially illegal traces) are from the legal situation. For medical staff these two situations can be perceived as qualitatively similar with a different degree of gravity. The notion of ∞ can be used to represent distance between traces perceived so qualitatively different to be incomparable. An example: a wrong action compromising the health of a patient takes place.*

We discuss here some concrete distances between the traces that will be useful in comparing tentative executions with the output of an enforcement mechanism.

Definition 3. *The* suppressing distance *between two finite traces is a total function $d_S : \Sigma^* \times \Sigma^* \to \mathbb{N} \cup \{\infty\}$, such that*

$$d_S(a\sigma, b\sigma') = \begin{cases} \infty & \text{if } a\sigma = \cdot \text{ and } b\sigma' \neq \cdot \\ |a\sigma| & \text{if } b\sigma' = \cdot \\ d_S(\sigma, \sigma') & \text{if } a = b \\ 1 + d_S(\sigma, b\sigma') & \text{if } a \neq b \end{cases} \tag{1}$$

Distance d_S counts number of actions that must be suppressed to obtain the second trace from the first one. The last rule says that if the first actions of the two sequences are different then the action in the first sequence is suppressed.

[1] The concepts of metrics and metric spaces are adapted from [1, 6, 15].

Reality Check 5 *The intuition behind suppression is that a bad trace is close to a good trace if the actions we have to undo are few. This can be explained to the operator and can be acceptable for an administrative procedure (albeit annoying for the operator involved). For example the monitor could block the process if the number of bad actions exceeds a given threshold or, preferably, it could undo all bad actions bringing us back to the point where we started the transaction that has gone awry.*

It is obvious that suppressing distance does not satisfy the symmetry property of metrics. Hence, the suppressing distance is not a metric but a *quasi-metric*.

Example 3. A doctor is selecting a drug (Dis) for which therapeutical notes are needed (Tnn). However at the time to review them, he is interrupted (e.g. in case of an emergency, interruption by another colleague etc.). When he comes back, he has to start running the process again because it timed out. The second time the doctor successfully finishes the process. So, the tentative execution is Dis; Tnn; NoteRun and the distance to the good trace NoteRun is 2. Notice that the distance from the good trace to the bad one is ∞ because no number of suppressions can transform the good trace into the bad trace.

This distance already discriminates between different run-time monitors:

Example 4. If the mechanism that is used to enforce this policy is a security automaton, it will stop executing the process as soon as something wrong happens (action Dis after Tnn in our case). Then, $E_{SA}(\mathsf{Dis};\mathsf{Tnn};\mathsf{NoteRun}) = \cdot$. If we use the suppressing distance to compare the outputs, $d_S(\cdot, E_{SA}(\mathsf{NoteRun})) = \infty$. The iterative suppression automaton [3] would have suppressed the initial prefix:

$$d_S(E_{IS}(\mathsf{Dis};\mathsf{Tnn};\mathsf{NoteRun}), E_{IS}(\mathsf{NoteRun})) = d_S(\mathsf{NoteRun},\mathsf{NoteRun}) = 0 \quad (2)$$

To compare outputs of more enforcement mechanisms we can use another metric that counts the number of replaced actions.

Reality Check 6 *While enforcing a process in the hospitals, an enforcement mechanism could be entitled only to correcting small errors without changing the protocol used by the operators (such as patient identification, patient consent and blood sampling before blood transfusion). If the doctor forgot to fill one field in the form, the mechanism can insert a default value. On the other hand, insertions of new steps by the monitor to compensate a bad event are not be allowed because a different protocol might have different medical or legal consequences and those can only be judged by an expert.*

Definition 4. *The* replacing distance *between two finite traces is a total function* $d_R : \Sigma^* \times \Sigma^* \to \mathbb{N} \cup \{\infty\}$*, such that*

$$d_R(a\sigma, b\sigma') = \begin{cases} 0 & \text{if } a\sigma = \cdot \text{ and } b\sigma' = \cdot \\ \infty & \text{if } a\sigma = \cdot \text{ xor } b\sigma' = \cdot \\ d_R(\sigma,\sigma') & \text{if } a = b \\ 1 + d_R(\sigma,\sigma') & \text{if } a \neq b \end{cases} \quad (3)$$

Distance d_R counts number of replacements to obtain one trace from another. If the traces have different length, the distance is ∞ (as expected as they clearly belong to different protocols). The replacing distance is a *metric*.

A more general definition of distance was originally proposed by Levenshtein [12]. This distance counts number of insertions, suppressions and replacements needed to obtain one trace from another.

Definition 5. *The* Levenshtein distance *between two finite traces is a total function* $d_L : \Sigma^* \times \Sigma^* \to \mathbb{N}$*, such that*

$$d_L(a\sigma, b\sigma') = \begin{cases} |b\sigma'| & \text{if } a\sigma = \cdot \\ |a\sigma| & \text{if } b\sigma' = \cdot \\ d_L(\sigma, \sigma') & \text{if } a = b \\ 1 + \min(d_L(\sigma, \sigma'), d_L(a\sigma, \sigma'), d_L(\sigma, b\sigma')) & \text{if } a \neq b \end{cases} \tag{4}$$

Let's make some examples of replacing and Levenshtein distances between the valid traces (Ex. 1) and the invalid ones (Ex. 2).

Example 5. The replacing distance can be equal to the Levenshtein distance: $d_R(\mathsf{NoteResearchRun}, \mathsf{CloseNoteProt}) = d_L(\mathsf{NoteResearchRun}, \mathsf{CloseNoteProt}) = 2$ since action Rtn is replaced with Ctw and action Irpn is replaced with Cpw. The replacing distance between CloseProt and NoteResearchRun is ∞ because NoteResearchRun is simply longer than CloseProt. But the Levenshtein distance counts the inserted and replaced actions and the distance is equal to 3.

5 From Sound to Bounded Monitors

Building on metrics over the sequences we propose to use the notions from measure theory. In the following definitions $(\Sigma^\infty \cup \mathrm{T}^\infty, d)$ is a metric space, a map $E : \Sigma^\infty \to \mathrm{T}^\infty$ is an enforcement mechanism and a security policy is a set $P \subseteq \Sigma^\infty \cap \mathrm{T}^\infty$. As a starting point we assume that all monitors in this paper are transparent.

Reality Check 7 *The actions in the systems corresponds to actions of doctors and nurses (or administrative staff) who are knowledgeable and accountable for their actions. Their point of view is that the choice of a* legitimate *course of action is due to a contextual knowledge not available to the system. A system that would change their decisions, when those actions conform to the policy of the hospital, would be unacceptable.*

The next step is generalizing the notion of soundness. We start from a classical definition of bounded map that Fig. 3 shows graphically. Even though the division of traces into good and bad is not relevant for the definition of boundedness, we keep it in the figure to ease the comparison with other notions.

Definition 6. *A function* $E : \Sigma^\infty \to \mathrm{T}^\infty$ *is* bounded *if the subset* $\{E(\sigma) : \sigma \in \Sigma^\infty\} \subseteq \mathrm{T}^\infty$ *is bounded. Formally,* $\exists \tau \in \mathrm{T}^\infty : \exists \varepsilon > 0 : \forall \sigma \in \Sigma^\infty : d(E(\sigma), \tau) \leq \varepsilon$.

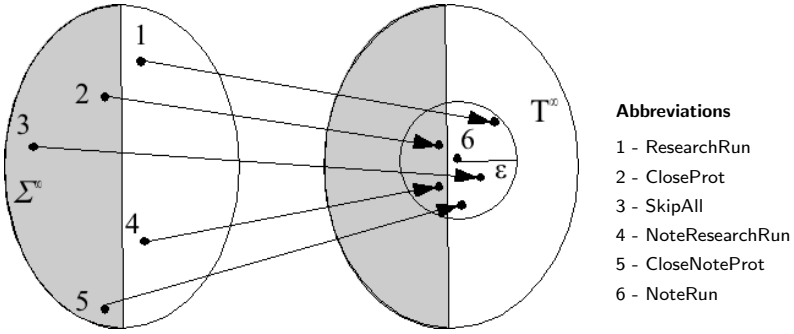

Fig. 3. Bounded map.

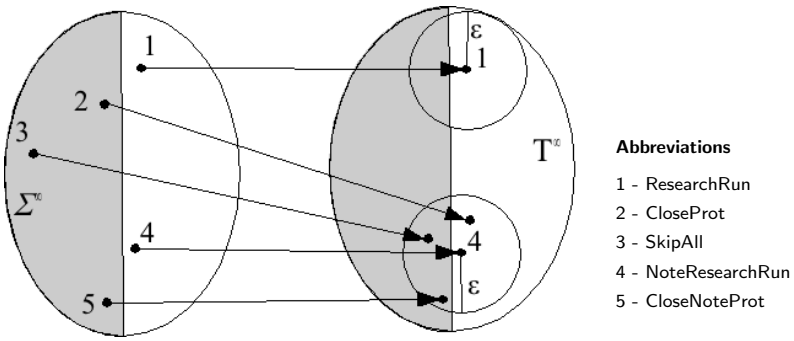

Fig. 4. Bounded within ε enforcement mechanism.

Let us project this notion to the theory of enforcement mechanisms. We get a mechanism that transforms all sequences to some sequence close to τ. This is not what users are expecting. The user's policy usually contains several good sequences (see Ex. 1). Hence we should adapt the definition to map bad sequences to different good sequences in the policy.

Definition 7. *An enforcement mechanism* $E : \Sigma^\infty \to \mathrm{T}^\infty$ *is* bounded within ε *iff* $\forall \sigma \in \Sigma^\infty : \exists \sigma_P \in P : d(E(\sigma), E(\sigma_P)) \leq \varepsilon$.

This notion says that an output of enforcement mechanism is always within the distance ε from some good execution, for $\varepsilon > 0$ we are weakening the notion of soundness. Fig. 4 shows the bounded within ε enforcement mechanism.

There is a fundamental difference between boundedness and boundedness within ε. Boundness means that there is one single trace τ in the possible outputs such that *all* the outputs of E are in the radius ε from τ. Boundedness within ε means that for every possible output there is a valid trace such that the distance between them is smaller or equal than ε.

Example 6. An enforcement mechanism E enforces the policy from Ex. 1 in the following way (we also show the distance to NoteResearchRun):

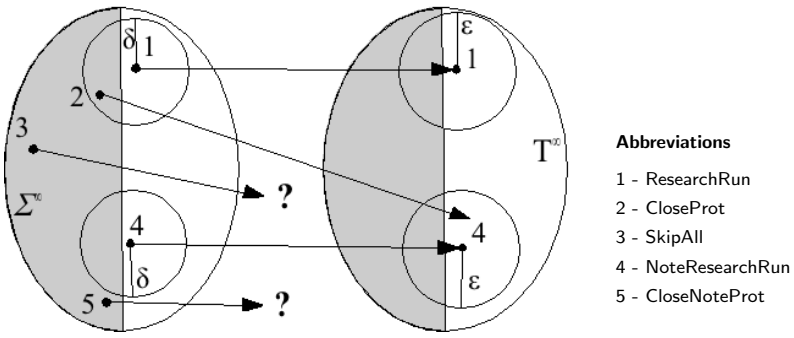

Fig. 5. Conditionally bounded within ε enforcement mechanism.

$E(\mathsf{CloseNoteProt}) = \mathsf{CloseNoteProt}$, $d_L(\mathsf{CloseNoteProt}, \mathsf{NoteResearchRun}) = 2$
$E(\mathsf{SkipAll}) = \mathsf{CloseNoteProt}$, $d_L(\mathsf{CloseNoteProt}, \mathsf{NoteResearchRun}) = 2$
$E(\mathsf{CloseProt}) = \mathsf{NoteRun}$, $d_L(\mathsf{NoteRun}, \mathsf{NoteResearchRun}) = 2$

Since the output of E is always at distance 2 from some good execution NoteResearchRun then E is bounded within $\varepsilon = 2$.

We can refine the notion by imposing also that "almost valid traces" should be mapped to some "almost valid traces". We call it *conditional boundedness within* ε and show graphically in Fig. 5.

Definition 8. *An enforcement mechanism* $E : \Sigma^\infty \to \mathrm{T}^\infty$ *is* conditionally bounded within ε *iff* $\exists \delta > 0 : \forall \sigma \in \Sigma^\infty : (\exists \sigma'_P \in P : d(\sigma, \sigma'_P) \leq \delta \Rightarrow \exists \sigma_P \in P : d(E(\sigma), E(\sigma_P)) \leq \varepsilon)$.

Example 7. An enforcement mechanism E transforms the sequences of actions in the following way (we also show the distance to some good sequence):

$E(\mathsf{CloseNoteProt}) = \mathsf{CloseProt}$, $d_L(\mathsf{CloseProt}, \mathsf{ResearchRun}) = 1$
$E(\mathsf{SkipAll}) = \mathsf{NoRun}$, $d_L(\mathsf{NoRun}, \mathsf{NoRun}) = 0$
$E(\mathsf{CloseProt}) = \mathsf{NoteRun}$, $d_L(\mathsf{NoteRun}, \mathsf{NoteRun}) = 0$

Obviously, E is bounded within $\varepsilon = 1$. It is conditionally bounded within $\varepsilon = 1$ because there is a $\delta = 3$ such that for every sequence there is a valid trace at most at distance 3: $d_L(\mathsf{CloseNoteProt}, \mathsf{NoteResearchRun}) = 2$, $d_L(\mathsf{SkipAll}, \mathsf{NoRun}) = 3$ and $d_L(\mathsf{CloseProt}, \mathsf{ResearchRun}) = 1$.

Reality Check 8 *Boundedness and conditional boundedness can refine soundness but are still unacceptable: if a doctor skips key steps altogether (SkipAll), a bounded EM can transform it into a sequence CloseNoteProt. A conditionally bounded, sound and transparent EM, when the doctor closes the window (CloseProt) instead of inserting the protocol (ResearchRun), can transforms this tentative execution into another, completely different one (NoteRun). The problem is that some actions in the outcome are not a direct transformation of the actions of the doctors. Since actions carry liabilities it is important that the actions of the EM are always linked to corresponding actions by the doctor.*

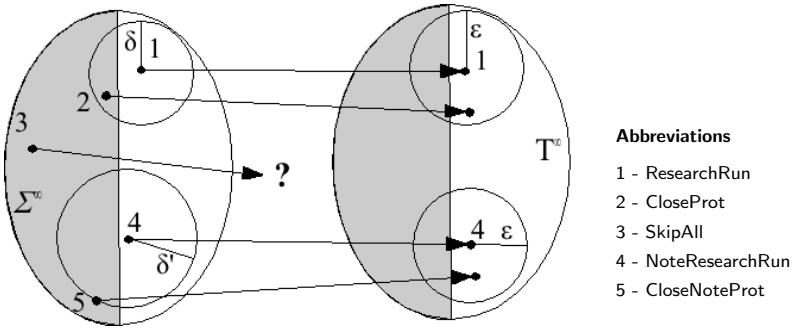

Fig. 6. Predictable within ε enforcement mechanism

6 Predictability

Our notion of predictability within ε is inspired by the classical notion of continuous functions. Let $(\Sigma^\infty \cup \mathrm{T}^\infty, d)$ and $(\Sigma^\infty \cup \mathrm{T}^\infty, d')$ be metric spaces.

Definition 9. *A map* $E : \Sigma^\infty \to \mathrm{T}^\infty$ *is* continuous *if at every trace* $\sigma \in \Sigma^\infty$ *the following holds:* $\forall \varepsilon > 0 : \exists \delta > 0 : \forall \sigma' \in \Sigma^\infty (d(\sigma, \sigma') < \delta \Rightarrow d'(E(\sigma), E(\sigma')) < \epsilon)$.

In conditional boundedness we proposed to limit an output when an input is "almost good". But as a Reality Check 8 shows, an input and its output should be compared to *the same* valid trace.

Definition 10. *An enforcement mechanism* E *is* predictable within ε *if for every trace* $\sigma_P \in P$ *the following holds:* $\forall \nu \geq \varepsilon : \exists \delta > 0 : \forall \sigma \in \Sigma^* : (d(\sigma, \sigma_P) \leq \delta \Rightarrow d'(E(\sigma), E(\sigma_P)) \leq \nu)$.

Informally, it says that for every valid trace there always exists a radius δ, such that all the traces within this radius are mapped into the circle with radius ε from this trace. We show it in Fig. 6.

Example 8. The enforcement mechanism E from Ex. 7 is conditionally bounded within $\varepsilon = 1$, but not predictable within $\varepsilon = 1$ since there exists $\sigma_P = \mathsf{ResearchRun}$ such that $\exists \nu = 2 : \forall \delta > 0 : \exists \sigma \in \Sigma^* : (d(\sigma, \sigma_P) \leq \delta \wedge d(E(\sigma), E(\sigma_P)) > \nu)$ where $\sigma = \mathsf{CloseProt}$, then $d(\mathsf{CloseProt}, \mathsf{ResearchRun}) = 1 \leq \delta$ and $d(E(\mathsf{CloseProt}), E(\mathsf{ResearchRun})) = d_L(\mathsf{NoteRun}, \mathsf{ResearchRun}) = 3 > 2$.

Table 2 shows all the notions so far and the new notion of predictability.

7 Conclusions

In this paper we have discussed how to go beyond the (only) two classical properties used to evaluate an enforcement mechanism: *soundness* and *transparency*. Soundness specifies that the output is always good and transparency guarantees that good input is not changed. However those two characteristics alone are not

Table 2. Properties of enforcement mechanism.

Name	Pre-Condition	Post-Condition
Soundness		for every trace the output is always some valid trace
Transparency	if input is a valid trace	output is *the same* valid trace
Bounded Map		there is one valid trace such that output is always within ε from that trace
Boundedness		output is always within ε from some valid trace
Conditional Boundedness	if input is within δ from some valid trace	output is within ε from some valid trace
Predictability	if input is within δ from a valid trace	output is within ε from *the same* valid trace

sufficient to discriminate between enforcement mechanisms. The key issue is to specify how bad input is fixed into good output.

We have introduced several notions that could describe predictable behavior and checked them against the industrial case study on e-Health. The idea behind *predictability* is that there are "no surprises on bad inputs".

An apparent limitation of our work is that we don't deal with infinite traces. We have actually considered some mathematical functions over infinite traces from [5] but we found a practical obstacle:

Reality Check 9 *The end of the fiscal year effectively terminates any trace in the eyes of our stakeholders.*

Further, all simple and natural metrics that we considered from [5] were not adequate from one perspective or another (we give some examples in the appendix).

Another open issue is the analysis of existing mechanisms to identify which one is predictable. The practically interesting question is whether a ε for predictability can be extracted from a security policy expressed as an automata.

The second issue revolves around edit automata as an enforcement mechanism and the characterization of predictable policies. A key question is whether policies of a certain form do always (or never) have predictable enforcement mechanism. Under some definition of convexity we could prove that convex policies always have predictable enforcement mechanisms within a bound fixed on the border, but the natural definition, while mathematically sound, is not intuitive enough to pass our reality check. More research is needed.

References

1. L.S. Pontryagin (eds.) A.V. Arkhangel'skii. *General topology I : basic concepts and constructions, dimension theory.* Springer-Verlag, 1990.
2. L. Bauer, J. Ligatti, and D. Walker. Edit automata: Enforcement mechanisms for run-time security policies. *Int. J. of Inform. Sec.*, 4(1-2):2–16, 2005.

3. N. Bielova, F. Massacci, and A. Micheletti. Towards practical enforcement theories. In *Proc. of The 14th Nordic Conference on Secure IT Systems*, volume 5838 of *LNCS*, pages 239–254. Springer-Verlag Heidelberg, 2009.
4. A. Brown and M. Ryan. Synthesising monitors from high-level policies for the safe execution of untrusted software. In *Proc. of the 4th Inf. Security Practice and Experience Conf.*, pages 233–247. Springer-Verlag Heidelberg, 2008.
5. K. Chatterjee, L. Doyen, and T. A. Henzinger. Expressiveness and closure properties for quantitative languages. *Comp. Research Repository*, abs/1007.4018, 2010.
6. D.L. Cohn. *Measure Theory*. Birkhauser, 1980.
7. L. Desmet, W. Joosen, F. Massacci, P. Philippaerts, F. Piessens, I. Siahaan, and D. Vanoverberghe. Security-by-contract on the .net platform. *Information Security Technical Report*, 13(1):25–32, 2008.
8. U. Erlingsson. *The Inlined Reference Monitor Approach to Security Policy Enforcement*. PhD thesis, Cornell University, 2003.
9. Y. Falcone, J.-C. Fernandez, and L. Mounier. Enforcement monitoring wrt. the safety-progress classification of properties. In *Proc. of 24th ACM Symp. on Applied Computing – Software Verif. and Test. Track*, pages 593–600. ACM Press, 2009.
10. G. Gheorghe, S. Neuhaus, and B. Crispo. xESB: An enterprise service bus for access and usage control policy enforcement. In *Proc. of the 4th IFIP WG 11.11 Int. Conf. on Trust Management*, volume 321, pages 63–78. Springer Boston, 2010.
11. R. Khoury and N. Tawbi. Using Equivalence Relations for Corrective Enforcement of Security Policies. In *Proc. of the 5th Int. Conf. on Math. Methods, Models, and Architectures for Comp. Network Sec.*, volume 6258, pages 139–154. Springer Berlin Heidelberg, 2010.
12. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966. An English translation of the "Physics Sections" of the Proceedings of the Academy of Sciences of the USSR.
13. J. Ligatti, L. Bauer, and D. Walker. Run-time enforcement of nonsafety policies. *ACM Trans. on Inform. and Sys. Security*, 12(3):1–41, 2009.
14. J. Ligatti and S. Reddy. A theory of runtime enforcement, with results. In *Proc. of ESORICS'10*, volume 6345, pages 87–100. Springer-Verlag Heidelberg, 2010.
15. S.G. Matthews. Partial metric topology. In *Proceedings of the 8th Summer Conference, Queen's College*, volume 728, pages 183–197. Annals of the New York Academy of Sciences, 1994.
16. J. Park and R. Sandhu. The UCON ABC usage control model. *ACM Trans. on Inform. and Sys. Security*, 7(1):128–174, 2004.
17. P.H. Phung, D. Sands, and A. Chudnov. Lightweight self-protecting javascript. In *Proc. of ACM Symp. on Inform. Comp. and Comm. Security*, pages 47–60. ACM Press, 2009.
18. A. Pretschner, M. Hilty, D. Basin, C. Schaefer, and T. Walter. Mechanisms for usage control. In *Proc. of ACM Symp. on Inform. Comp. and Comm. Security*, pages 240–244. ACM Press, 2008.
19. F.B. Schneider. Enforceable security policies. *ACM Trans. on Inform. and Sys. Security*, 3(1):30–50, 2000.
20. C. Talhi, N. Tawbi, and M. Debbabi. Execution monitoring enforcement under memory-limitation constraints. *Inform. and Comp.*, 206(2-4):158–184, 2007.
21. D. Yun, A. Chander, N. Islam, and I. Serikov. Javascript instrumentation for browser security. In *Proc. of the 34th ACM SIGPLAN-SIGACT Symp. on Princ. of Prog. Lang.*, pages 237–249. ACM Press, 2007.

A Examples of Levenshtein distances

Table 3. The Levenshtein distances between some sequences

	NoRun	ResearchRun	CloseProt	SkipAll	NoteResearchRun	CloseNoteProt
NoRun	0	-	-	-	-	-
ResearchRun	5	0	-	-	-	-
CloseProt	5	1	0	-	-	-
SkipAll	3	2	2	0	-	-
NoteResearchRun	6	2	3	3	0	-
CloseNoteProt	6	3	2	3	2	0
NoteRun	5	3	3	3	2	3

B Examples of (unsatisfactory) Metrics on Infinite Traces

Here we discuss two natural distances from [5] that have a clear mathematical intuition for our domain and allow to obtain finite numbers when comparing infinite traces. The first option is to use an economic approach and consider the discounted distance that discounts the remote differences in the sequence :

Definition 11. *The* discounted distance *between two infinite traces is a total function* $d_D : \Sigma^\omega \times \Sigma^\omega \to \mathbb{N}$*, such that*

$$d_D(a\sigma, b\sigma') = \begin{cases} |a\sigma| & \text{if } b\sigma' = \cdot \\ |b\sigma'| & \text{if } a\sigma = \cdot \\ d_D(\sigma, \sigma') & \text{if } a = b \\ 1 + \frac{1}{k}\, d_D(\sigma, \sigma') & \text{if } a \neq b \end{cases} \tag{5}$$

In this definition $k \in \mathbb{R}$, $k \neq 0$ is a discounting factor.

Reality Check 10 *The first function can be acceptable from the perspective of a risk manager as later events have less risk of being detected or of having consequences within the year. It is less acceptable for doctors: a wrong drug is a wrong drug, no matter if delivered at the beginning or the end of the fiscal year.*

Another approach that works for replacing-type distances is to attribute a weight to each replacement and consider the maximum of such weights.

Reality Check 11 *The maximum weight determines what can at worst happen and can be satisfactory from the point of view of an operator: at worst it deviated so and so from the ideal trace. It is not satisfactory from a risk management perspective, as it cannot distinguish between a trace where such deviations are rare from trace where they are frequent.*

At the end the only acceptable metrics boils down to considering what happens during a limited slot and project it to infinity. But then we could simply consider the finite slot.

Hence, the definition of mathematically simple and meaningful metrics for infinite traces is still open for us (assuming we should consider them at all).