Cost-Effective Enforcement of UCON$_A$ Policies

Leanid Krautsevich
Department of Computer Science
University of Pisa
Largo B. Pontecorvo 3, Pisa, Italy
Email: krautsev@di.unipi.it

Aliaksandr Lazouski, Fabio Martinelli, Artsiom Yautsiukhin
Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche
G. Moruzzi 1, Pisa, Italy
Email: aliaksandr.lazouski@iit.cnr.it
fabio.martinelli@iit.cnr.it
artsiom.yautsiukhin@iit.cnr.it

***Abstract*—In Usage CONtrol (UCON) access decisions rely on mutable attributes. A reference monitor should re-evaluate security policies each time when attributes change their values. Catching timely all attribute changes is a challenging issue, especially if the attribute provider and the reference monitor reside in different security domains. Some attribute changes might be missed, corrupted, and delayed. As a result, the reference monitor may erroneously grant the access to malicious users and forbid it for eligible users.**

This paper proposes a set of policy enforcement models which help to tolerate uncertainties associated with mutable attributes. In our model the reference monitor as usually evaluates logical predicates over attributes and additionally makes some estimates on how much observed attribute values differ from the real state of the world. The final access decision counts both factors. We assign monetary outcomes for granting and revoking access to legitimate and malicious users and compare the proposed policy enforcement models in terms of cost-efficiency.

***Index Terms*—Usage Control, Mutable Attribute, Policy Enforcement, Cost, Markov Chain**

I. Introduction

Access control aims to assure that only trusted principals are granted to access a computational resource [1]. *Usage control* is in charge to guarantee that principals remain trusted also when the access is in progress, i.e. when these principals use the resource. The principal's trustworthiness is evaluated by the reference monitor based on security attributes [13], [14], assertions done by the attribute provider about subjects and objects participating in access and usage control.

The UCON model proposed by R. Sandhu et al. [17] encompasses access and usage control scenarios and operates with mutable attributes to specify and enforce security policies. Access decisions in UCON are based on authorizations (predicates over subject and object attributes), conditions (predicates over environmental attributes), and obligations (actions that must be performed by a requesting subject). The reference monitor in UCON re-evaluates the access each time when an attribute changes its value. Catching timely all attribute changes is a challenging issue.

The nature of security attributes is diverse and some attributes (e.g., the requester's reputation and location) are *remote*, reside outside the control of the reference monitor, and can be only observed. These attributes should be constantly *pushed* by the attribute provider (e.g. the requester) or *pulled* by the reference monitor. The system usually allows pulling only of the current attribute value, and as a result some attribute changes between adjacent pulling queries might be missed. Worse, these unnoticed changes might violate security policies. For example, if a security policy grants access rights to users resided in a certain location, there is no evidence that mobile users remained in the same location and never was leaving it in-between checks [5].

Also, a system faultiness, delays occurred during attributes delivery due to the network latency, and malicious activities (e.g., a man-in-the-middle, eavesdropping and impersonating of data by the attribute provider) contribute to the problem of correct policy enforcement. The impact of uncertainties associated with observed attributes should be tolerated by the reference monitor.

This paper proposes the cost-effective enforcement models of $UCON_A$ [17] security policies. Our basic idea is the following:

1) The reference monitor evaluates security policies with respect to observed attribute values;
2) If policies hold, the reference monitor runs an experiment which estimates how observed attributes differ from the real state of the world. If this difference is negligible, the experiment succeeds and the reference monitor grants (or continues) the access.

We assign monetary outcomes for granting and revoking access to legitimate and malicious users and compare the proposed policy enforcement models in terms of cost-efficiency.

The main contributions of this paper are:

- identifying and estimating the impact of all uncertainties associated with attributes used to produce access decisions;
- introducing models of a correct policy enforcement and enforcement under uncertainties;
- introducing a cost model for a policy enforcement and comparing the cost-efficiency of proposed enforcement models.

The paper is structured as follows. Section II gives basic notes on UCON. Section III introduces the model of a mutable attribute, and enlists all types of uncertainties associated with mutable attributes. Sections IV and V present models of correct policy enforcement and policy enforcement under uncertainties. Section VI outlines a cost model and estimates an

This work was partly supported by the EU-FP7-ICT NESSoS project.

978-1-4577-1891-5/11/$26.00 ©2011 IEEE

average profit of policy enforcement. Section VII summarizes related works. Section VIII concludes the paper.

II. Usage CONtrol

Usage control (UCON) [17] demands for continuous control over long-standing accesses to computational resources (e.g., an execution of a job in Grid, a run of a virtual machine in Cloud). Continuity of control is a specific feature of UCON intended to operate in an inconstant context. The context is formed by attributes of a requesting subject, an accessed object and execution environment.

An attribute is denoted as a variable $h.v$ where h identifies a subject requesting an object, the object itself, or environment, and v refers to the attribute name. An assignment of an attribute maps its name to a value in its domain Ω_{attr}, i.e., $h.v = r$, where $r \in \Omega_{attr}$. Without loss of generality, we assume that there is only one attribute in the system denoted as v and it assumes values from a finite domain.

Attribute mutability is an important feature of UCON. It means that an attribute can change its value as a result of an access request or caused by other uncontrollable factors. A *behaviour of an attribute* is simply a sequence of values assigned to attribute with time passage: $\{v_0, v_1, ..., v_i, ...\}$, where v_0 refers to the attribute value when a subject sends an access request. An index $i \in \mathbb{N}^0$ refers to a time point at which an attribute changes its value. We define a strictly increasing function cl which assigns a real time value to any index, that is, $cl : \mathbb{N}^0 \to \mathbb{R}$.

Access decisions in UCON are based on authorizations (predicates over subject and object attributes), conditions (predicates over environmental attributes), and obligations (actions that must be performed by a requesting subject). For the sake of simplicity, we consider security policies consisting of authorization predicates only, that is, the $UCON_A$ model. We define a predicate p to be a boolean-valued computable function mapping an attribute value to either true or false, $p : \Omega_{attr} \to \{true, false\}$.

Another important feature of UCON is that it assumes to specify when access decisions are evaluated and enforced. There are two phases called a pre-authorization, or *access control*, and a continuous policy enforcement, or *usage control*.

Access control phase is started by the reference monitor upon receiving a request at time t_{try}. The reference monitor acquires v_i, evaluates authorization predicates only once and grants access to a resource at time t_{perm} if $p(v_i) = true$, where $cl(i) \leq t_{perm} < cl(i+1)$.

Usage control phase is started by the reference monitor at time t_{perm}. The reference monitor continuously evaluates authorization predicates when the access is in progress. Continuity of control means that the reference monitor re-evaluates authorization predicates each time when an attribute changes its value. If a new value v_j violates a security policy, the reference monitor revokes the access. The access should be continued only if $p(v_{i+1}) \wedge p(v_{i+2}) \wedge ... \wedge p(v_j) = true$. Though usage control ends as a result of the access revocation or at subject's discretion, we consider only the first scenario. Usage control is over at time $t_{rev} = cl(k)$ if $p(v_k) = false$.

III. Attribute Model

Access decisions rely on correct attribute values. Our main concern in this paper is enforcement of a UCON policy based on a *remote* attribute with *observable mutability*. *Remote* means that an attribute is managed by the attribute provider which is not under control of the reference monitor. Hence, the reference monitor should trust the attribute provider to use this attribute. *Observable mutability* means that the reference monitor observes only partially how the attribute behaves in time. Thus, for the same attribute we distinguish *real attribute values* which truly describe the attribute behaviour in the system and *observed attribute values* which are obtained by the reference monitor and used to evaluate authorization predicates.

A. Real Attribute Values

Since an attribute in UCON is mutable and changes its value as a result of the access or caused by other uncontrollable factors, we introduce a random variable A which models the attribute value at some point of time. A is a real valued function on the attribute's domain Ω_{attr}, that is, $A : \Omega_{attr} \to \mathbb{R}$. The event "$A = a$" represents the fact that the attribute value is a. Let probability of that event to happen be $\mathbf{Pr}(A = a)$. The function $\mathbf{Pr}$ has all properties of a probability function, e.g., for any event E, $0 \leq \mathbf{Pr}(E) \leq 1$. Let event $\mathcal{P}(A)$ denote that an attribute value satisfies authorization predicates, i.e., $\mathcal{P}(A) : A = a$, $p(a) = true$, and assume the event $\overline{\mathcal{P}}(A)$ specifies the opposite, i.e. the attribute value violates the predicate.

Let a behaviour of a *real attribute* be specified by a scheme $\langle \mathbf{A}, \mathbf{CL}_{AP} \rangle$, where:

- $\mathbf{A} = \{A_i : i \in \mathbb{N}^0\}$ is a discrete-time stochastic process modelling a mutable attribute. We call A_i the state of the process at i, and $A_i = a$ denotes that after i changes, the attribute value equals to a;
- $\mathbf{CL}_{AP} = \{cl_{AP}(j) | j \in \mathbb{N}^0\}$ is an ordered set of timestamps assigned to each attribute change by the attribute provider when it happens. We assume that $cl_{AP}(0) = t_{try}$ and for all $j \geq 1$, $cl_{AP}(i) = cl_{AP}(i-1) + \mathrm{T}$. T is either a constant or a random variable specifying a time interval between adjacent attribute changes.

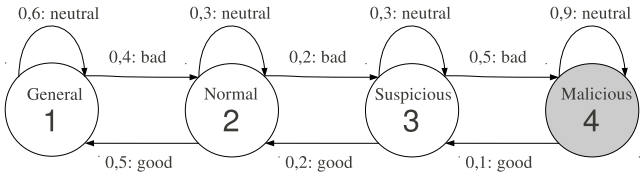

Fig. 1. A Reputation Attribute Model

Example 1: Following the approach in [10], [9] we consider a mutable attribute which encodes a reputation of a requester. The attribute domain is $\Omega_{attr} = \{$"general", "normal",

"suspicious", "malicious"} and respectively A("general") = 1, A("normal") = 2, A("suspicious") = 3, and A("malicious") = 4.

The attribute value is changed based on "bad", "good" and "neutral" feedback received from other parties (see Figure 1). The attribute mutability is modelled as a discrete-time Markov chain $\mathbf{A}$ uniquely defined by the one-step transition matrix:

$$\mathbf{Prob} = \begin{pmatrix} 0.6 & 0.4 & 0.0 & 0.0 \\ 0.5 & 0.3 & 0.2 & 0.0 \\ 0.0 & 0.2 & 0.3 & 0.5 \\ 0.0 & 0.0 & 0.1 & 0.9 \end{pmatrix} \quad (1)$$

That is, the entry in the i-th row and j-th column is the transition probability $\mathbf{Pr}(A_i = a \mid A_{i-1} = b)$, giving the probability that the attribute will change value to a if its current value is b.

A time interval between adjacent attribute changes can be modelled as a constant with rate *2 seconds*.

If the initial attribute value was $A_0 = 3$, the possible attribute behaviour could be:

$$(A_0 = 3 : 0s), (A_1 = 4 : 2s), (A_2 = 3 : 4s), (A_3 = 2 : 6s), ...$$

B. Observed Attribute Values

Only the attribute provider knows how the attribute behaves in time, but the reference monitor can also observe this process. There are two basic models how attribute changes are delivered to the reference monitor: *push* and *pull*. Push model defines a scenario when each new attribute value is timestamped and pushed by the attribute provider to the reference monitor. Pull model defines a scenario when the reference monitor queries the attribute provider to give the current attribute value, the attribute provider replies with the value, its timestamp and some additional information.

By analogy with real attribute values, let *observed attributes* be specified by a scheme $\langle \tilde{\mathbf{A}}, \mathbf{CL}_{RM} \rangle$, where:

- $\tilde{\mathbf{A}} = \{\tilde{A}_i : i \in \mathbb{N}^0\}$ is a discrete-time stochastic process modelling an observation of attribute changes over time. $\tilde{A}_i = a$ denotes that an attribute value after i observations equals to a;
- $\mathbf{CL}_{RM} = \{cl_{RM}(j) | j \in \mathbb{N}^0\}$ is an ordered set of timestamps which assigned by the reference monitor. A timestamp j denotes when the j-th observation of an attribute value was processed and the appropriate access decision was enforced by the reference monitor. We assume that $cl_{RM}(0) = t_{perm}$.

Real and observed attribute values form a bipartite directed graph $\mathrm{W} = (\mathbf{A}, \tilde{\mathbf{A}}, \mathbf{E})$, where edges $\mathbf{E}$ connect real and observed attributes via push/pull queries(see Figure 2). If there exists an edge e which connects A_c and $\tilde{A}_{c'}$, we say that A_c corresponds to $\tilde{A}_{c'}$ and denote this as $A_c \approx \tilde{A}_{c'}$. To evaluate authorization predicates, the reference monitor can exploit observed attribute values and timestamps of the corresponding real counterparts.

C. Intentional and Unintentional Uncertainties

Observed attributes vary from the real counterparts due to attacks, noise, delays during delivery, missed attributes, etc. We call *uncertainty* a property on real and observed attributes which specifies how these attributes vary. As closer observed attributes to real, the better the policy is enforced. We launch two types of uncertainties: *unintentional* (*freshness* and *correctness*), and *intentional* (*trustworthiness*).

1) Freshness of Attributes: is unintentional uncertainty occurring due to attributes mutability. Generally, it means that the last observed value of an attribute is out-of-date, while the current real value of the attribute needed to produce the access decision is unknown. We introduce three types of freshness uncertainties.

Freshness I (missed attributes) corresponds to the scenarios where only a part of attribute changes is detected in observed attribute values:

$$\exists c \geq 0,\ m > 0\ :\ A_{c+m} \approx \tilde{A}_c$$

As an example, assume the network of sensors providing the current location of the user. Sensors have limited resources (power, bandwidth, memory), and the reference monitor pulls the location attribute only once per hour. Even if the attribute does not satisfy the policy during this hour, the reference monitor will make the incorrect access decision and continue the access. There always exists a possibility of the policy violation in-between despite that all observed attribute changes satisfy the policy.

Freshness II (delays in processing) implies that there are inevitable time delays needed for delivery of an attribute (due to a network latency) and decision making (evaluation of authorization predicates). That is:

$$\exists c' \geq 0,\ c'' \geq 0\ :\ A_{c''} \approx \tilde{A}_{c'}$$
$$cl_{RM}(c') > cl_{AP}(c'')$$

Freshness III (pending updates) corresponds to scenarios where the current attribute value is uncertain since some update queries are pending at the time of the access re-evaluation. In this case, the attribute provider sends two values: (i) the last certain attribute value and, (ii) some additional information on how the real value varies from the last certain.

As an example, assume a policy which allows users with a "normal" reputation (see Example 1) to submit a huge number of applications for execution in Grid. The reputation is updated only when the execution is ended and the system receives feedback from a resource provider. Applications can run concurrently and each single execution can be long-lived and lasts days. The access decision to submit a new job is based on the reputation value dated by last registered feedback and on the number of applications currently running on the user's behalf. Indeed, the ongoing applications can be malicious but this fact can be discovered afterwards. The only way to obtain the fresh reputation value is to block the access until all running applications terminate. Instead, the system has

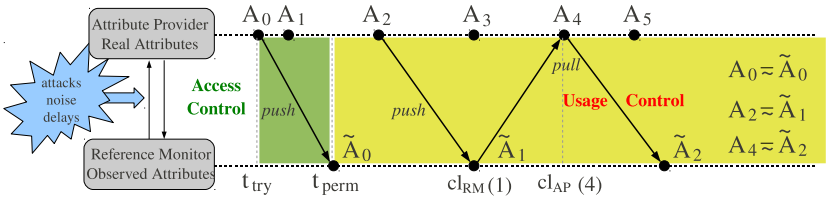

Fig. 2. Security Policy Enforcement with Pull Acquisition Model

to be set up to make an access decision with some uncertainty on the current reputation of the user.

The presence of the uncertainty freshness III implies:

$$\exists c' \geq 0,\ c'' \geq 0,\ m > 0\ :\ A_{c''} \approx \tilde{A}_{c'}$$
$$cl_{AP}(c'' + m) \leq cl_{RM}(c')$$

and the reference monitor knows m value

2) Correctness: is affected by additive noises that usually exist in case of non-accurate measurements. For example, the location attribute can be sensed only with the given precision. Thus, observed attribute values differ from the real counterparts:

$$\exists c' \geq 0,\ c'' \geq 0\ :\ A_{c''} \approx \tilde{A}_{c'}$$
$$\tilde{A}_{c'} = A_{c''} + N$$

and N is a random variable that models additive noises presented in observed attribute values.

3) Trustworthiness : appears as a result of altering attributes by the attribute provider or as the result of attacks occurred during attributes delivery, storing, etc. Current approaches guarantee only integrity of an attribute by validating a signature of the entity which signs the attribute, but this does not guarantee trustworthiness.

This uncertainty assumes that either an attribute value, or a time of issuance, or both can be modified by the attribute provider. Indeed, on each attribute request the attribute provider responds with the same value which always satisfies the policy. The presence of the trustworthiness uncertainty means that:

$$\forall c \geq 0 : \mathbf{Pr}[\mathcal{P}(\tilde{A}_c)] = const$$

i.e., the probability that the observed attribute satisfies authorization predicates remains constant and does not depend on how the attribute behaves in reality.

IV. Correct Policy Enforcement

The correct policy enforcement implies that having observed attributes the reference monitor enforces the policy exactly in the same fashion as with real attributes, and both observed and real attributes respect authorization predicates.

A. Correct Enforcement of Access Control

Access control starts at time $t_{try} = cl_{AP}(0)$ upon receiving the access request and the initial attribute value. The reference monitor evaluates authorization predicates only once and grants access to a resource at time $t_{perm} = cl_{RM}(0)$ if a policy holds. We say that the *policy holds for access control* if:

1) $p(\tilde{A}_0) = true$, i.e. the initial observed attribute value $\tilde{A}_0$ satisfies authorization predicates;
2) $p(A_m) = true$, i.e. the real attribute value A_m satisfies authorization predicates and $cl_{AP}(m) \leq t_{perm} < cl_{AP}(m+1)$ where $m \geq 0$;

Notice, some attribute changes may happen between t_{try} and t_{perm}, but attribute values must satisfy a security policy exactly when the request is issued and later, when the access decision is evaluated.

Let H be an event specifying that the policy holds and $\overline{H}$ specifies the opposite. Clearly, the policy satisfaction and violation can be also defined as:

$$H = \mathcal{P}(\tilde{A}_0) \cap \mathcal{P}(A_m) \qquad (2)$$
$$\overline{H} = \overline{\mathcal{P}}(\tilde{A}_0) \cup (\mathcal{P}(\tilde{A}_0) \cap \overline{\mathcal{P}}(A_m))$$

Where we write $E_1 \cap E_2$ for occurrence of both E_1 and E_2 and write $E_1 \cup E_2$ for the occurrence of either E_1 or E_2 (or both).

Definition 1: ***(Correct Enforcement of Access Control)*** The reference monitor grants the access at t_{perm} if the policy holds and denies otherwise.

Let G be an event specifying that the reference monitor grants the access and $\overline{G}$ specifies the opposite (denies the access). From Definition 1 we receive for the correct enforcement of access control:

$$G = H,\ \ \overline{G} = \overline{H} \qquad (3)$$

B. Correct Enforcement of Usage Control

Usage control phase is started at t_{perm}. The reference monitor re-evaluates authorization predicates each time when it observes an attribute change.

We say that a *policy holds for usage control* on a time interval $(t_b : t_e]$ if:

1) $p(\tilde{A}_k) \wedge p(\tilde{A}_{k+1}) \wedge ... \wedge p(\tilde{A}_l) = true$, where $t_b < cl_{RM}(k) < ... < cl_{RM}(l) \leq t_e$;
2) $p(A_i) \wedge p(A_{i+1}) \wedge ... \wedge p(A_j) = true$, where $t_b < cl_{AP}(i) < ... < cl_{AP}(j) \leq t_e$,

i.e., all real and observed attribute changes, occurred within this interval, do satisfy authorization predicates.

If there exists at least one attribute value (either real or observed) which does not satisfy authorization predicates, we call this as a *policy violation for usage control*. Let a *policy*

violation time refer to the interval during which real attribute values do not satisfy authorization predicates.

*Definition 2: (**Correct Enforcement of Usage Control**)* The reference monitor correctly continues the usage session at t_{now} if a policy holds on interval $(t_{perm} : t_{now}]$. The reference monitor revokes the access immediately when the policy violation happens, hence the policy violation time equals to nil.

V. Policy Enforcement under Uncertainties

Correct enforcement is not feasible in presence of uncertainties since the reference monitor is unable to show that real attribute values satisfy authorization predicates.

The basic idea for policy enforcement under uncertainties is as follows:

1) The reference monitor evaluates authorization predicates with respect to observed attribute values;
2) If so, the reference monitor runs an experiment which estimates how far are observed attributes from real counterparts. If this difference is negligible, the experiment succeeds and the reference monitor grants (or continues) the access.

A. Enforcement of Access Control

When uncertainties are present, we suppose that the reference monitor is powerful to mine some probabilistic knowledge about a real attribute changes based on the observed attribute $\tilde{A}_0 = a$:

$$\mathbf{Pr}_{RM} = \mathbf{Pr}[\mathcal{P}(A_m)|\tilde{A}_0 = a]$$

specifies conditional probability that a value of a real attribute A_m satisfies authorization predicates at time t_{perm} if the observed attribute value at time t_{perm} equals to a.

Let Y be a random variable such that:

$$Y = \begin{cases} 1 & \text{if uncertainties are negligible} \\ 0 & \text{otherwise} \end{cases}$$

Let $\delta(x)$ be a function, that is:

$$\delta(x) = \begin{cases} 1 & \text{if } x \geq th \\ 0 & \text{otherwise} \end{cases}$$

We propose two models of enforcement for access control under uncertainties, a *threshold* enforcement and a *flip coin* enforcement. The reference monitor is free to pick any of these models.

*Definition 3: (**Threshold Enforcement of Access Control**)* The reference monitor computes $\mathbf{Pr}_{RM}$ and grants the access at t_{perm} if:

1) $p(\tilde{A}_0) = true$;
2) $Y = 1$, where $\mathbf{Pr}[Y = 1] = \delta(\mathbf{Pr}_{RM})$,

otherwise, the access is denied.

That is, if the initial observed attribute value satisfies authorization predicates, the reference monitor grants the access if the probability that the real attribute value A_m also satisfies authorization predicates is above a specified threshold th.

*Definition 4: (**Flip Coin Enforcement of Access Control**)* The reference monitor behaves exactly as in the threshold enforcement but uses $\mathbf{Pr}[Y = 1] = \mathbf{Pr}_{RM}$ instead.

Hence, if the initial observed attribute value satisfies authorization predicates, the reference monitor runs the random experiment that succeeds (returns grant) with probability $\mathbf{Pr}_{RM}$ and fails (returns deny) with probability $1 - \mathbf{Pr}_{RM}$.

In notation of events, we get for the enforcement of access control under uncertainties (either threshold or flip coin):

$$\begin{aligned} G &= \mathcal{P}(\tilde{A}_0) \cap [Y = 1] \\ \overline{G} &= \overline{\mathcal{P}}(\tilde{A}_0) \cup (\mathcal{P}(\tilde{A}_0) \cap [Y = 0]) \end{aligned} \tag{4}$$

B. Enforcement of Usage Control

We call *check* a time interval $(t_b : t_e]$ where there is only one observed attribute value $\tilde{A}_k$, and $cl_{RM}(k) = t_e$.

By analogy with access control, we assume that the reference monitor is powerful to compute probability that a policy holds for usage control on any check:

$$\mathbf{Pr}^k_{RM} = \mathbf{Pr}[\mathcal{P}(A_i) \cap ... \cap \mathcal{P}(A_j)|\tilde{A}_k = a]$$

where $t_b < cl_{AP}(i) < ... < cl_{AP}(j) \leq t_e$

*Definition 5: (**Threshold Enforcement of Usage Control under Uncertainties**)* The reference monitor continues the access after n policy checks at $t_{now} = cl_{RM}(\tilde{A}_n)$ if:

1) $p(\tilde{A}_0) \wedge p(\tilde{A}_1) \wedge ... \wedge p(\tilde{A}_n) = true$, i.e., all observed attribute changes, occurred within n checks, do satisfy authorization predicates;
2) $\forall k = 0, .., n : Y_k = 1$, where $\mathbf{Pr}[Y_k = 1] = \delta(\mathbf{Pr}^k_{RM})$, i.e., for each check the probability that a policy holds on this check should be above a specified threshold;

otherwise, revokes the access.

*Definition 6: (**Flip Coin Enforcement of Usage Control under Uncertainties**)* The reference monitor behaves exactly as in the threshold enforcement but uses $\mathbf{Pr}[Y_k = 1] = \mathbf{Pr}^k_{RM}$ instead.

VI. Cost Model of the Policy Enforcement

We assign monetary outcomes for granting and revoking access to legitimate users and those whose attributes violate security policies. We estimate an *expected profit* $\langle C \rangle$ for enforcement of access and usage control.

A. Cost Matrix

The reference monitor chooses between two *alternatives* (grant access and deny/revoke access) only one, which is as good as possible. Good means that the reference monitor grants access to legitimate users and the policy holds, and forbids the access to unauthorized entities otherwise. In the presence of uncertainties, the reference monitor is unable to infer accurately whether the policy holds, and, consequently, to choose a good alternative. Mistakes are possible. There are four scenarios (events) how the reference monitor acts under uncertainties:

- $G \cap H$ *true positive*: grant access when policy holds;
- $G \cap \overline{H}$ *false negative*: grant access when policy is violated;

- $\overline{G} \cap H$ *false positive*: deny access when policy holds;
- $\overline{G} \cap \overline{H}$ *true negative*: deny access when policy is violated.

True positive and *true negative* are good-chosen alternatives, while *false negative* and *false positive* are erroneous. Each scenario has a monetary outcome, i.e. *cost*, the reference monitor loses/gains if this scenario happens.

Let C_{tp} denote a cost of the true positive scenario, when the reference monitor grants the access operating with observed attributes and the policy really holds. C_{fn}, C_{fp}, C_{tn} are costs of the remaining scenarios, respectively. The semantics of costs for access control corresponds to "pay-per-access" attributes, and specifies exact benefits and losses the system gains for a given access request. It is difficult to determine costs for every policy, but if the reference monitor behaves correctly the costs should be positive, i.e. $C_{tp} \geq 0, C_{tn} \geq 0$, and negative in the case of erroneous decisions, that is, $C_{fp}^{ac} < 0, C_{fn}^{ac} < 0$. Finally, let C_a be a cost to push/pull (observe) an attribute value.

B. Cost of Access Control Enforcement

The expected profit received by the reference monitor processing a single access request will be the summation of probabilities of all 4 scenarios weighted on corresponding costs. Without loss of generality, we assume that $C_{tn}^{ac} = 0$ and add to the expected profit a cost paid to observe the initial attribute value:

$$\begin{aligned}\langle C \rangle = C_{tp} \cdot \mathbf{Pr}[G \cap H] + C_{fn} \cdot \mathbf{Pr}[G \cap \overline{H}] \\ + C_{fp} \cdot \mathbf{Pr}[\overline{G} \cap H] + C_a\end{aligned} \tag{5}$$

1) Correct Enforcement: Since H and $\overline{H}$ are disjoint events, i.e. $\mathbf{Pr}[H \cap \overline{H}] = 0$ and $\mathbf{Pr}[H] + \mathbf{Pr}[\overline{H}] = 1$, from Equations 3 and 5 we receive:

$$\langle C \rangle_{cor} = C_{tp} \cdot \mathbf{Pr}[H] + C_a$$

Together with Equation 2 this yields the following average profit per access request for correct enforcement of access control:

$$\begin{aligned}\langle C \rangle_{cor} = C_{tp} \cdot \mathbf{Pr}[\mathcal{P}(\tilde{A}_0) \cap \mathcal{P}(A_m)] + C_a \\ = C_{tp} \cdot \mathbf{Pr}[\mathcal{P}(\tilde{A}_0)] \cdot \mathbf{Pr}[\mathcal{P}(A_m) | \mathcal{P}(\tilde{A}_0)] + C_a\end{aligned} \tag{6}$$

In what follows, we use $\mathbf{Pr}[\mathcal{P}(\tilde{A}_0)]$ interchangeably with α, and $\mathbf{Pr}[\mathcal{P}(A_m)|\mathcal{P}(\tilde{A}_0)]$ we encode with β.

2) Threshold Enforcement: We recall some properties of conditional probability:

$$\mathbf{Pr}[E_1 \cap E_2 \cap E_3] = \mathbf{Pr}[E_1] \cdot \mathbf{Pr}[E_2|E_1] \cdot \mathbf{Pr}[E_3|E_1 \cap E_2] \tag{7}$$

We also need to point out that probability of a policy satisfaction for real attributes is conditionally independent of estimates of the reference monitor given that observed attribute values satisfy the policy. That is:

$$\mathbf{Pr}[\mathcal{P}(A_m)|Y = 1 \cap \mathcal{P}(\tilde{A}_0)] = \mathbf{Pr}[\mathcal{P}(A_m)|\mathcal{P}(\tilde{A}_0)] \tag{8}$$

From Equations 2, 4, 7 and 8 we receive:

$$\begin{aligned}\mathbf{Pr}[G \cap H] &= \alpha \cdot \beta \cdot \mathbf{Pr}[Y = 1|\mathcal{P}(\tilde{A}_0)] \\ \mathbf{Pr}[G \cap \overline{H}] &= \alpha \cdot (1 - \beta) \cdot \mathbf{Pr}[Y = 1|\mathcal{P}(\tilde{A}_0)] \\ \mathbf{Pr}[\overline{G} \cap H] &= \alpha \cdot \beta \cdot (1 - \mathbf{Pr}[Y = 1|\mathcal{P}(\tilde{A}_0)])\end{aligned} \tag{9}$$

Without loss of generality, we assume that all access requests come with the identical initial value of the attribute and let a be this value which satisfies authorization predicates. With this assumption, we get that $\alpha = 1$, and $\mathbf{Pr}[Y = 1|\mathcal{P}(\tilde{A}_0)] = \mathbf{Pr}[Y = 1|\tilde{A}_0 = a]$

From Definition 3 and Equations 9 and 5 we get that the average profit for a threshold enforcement of access control is given by:

$$\langle C \rangle_{th} = \begin{cases} \beta \cdot (C_{tp} - C_{fn}) + C_{fn} + C_a & \text{if } \mathbf{Pr}_{RM} \geq th \\ \beta \cdot C_{fp} + C_a & \text{otherwise} \end{cases}$$

Obviously, the minimal average cost for the threshold enforcement is the following:

$$\beta_{min} = \frac{C_{fn}}{C_{fp} + C_{fn} - C_{tp}} \tag{10}$$

This means that when a "distance" between real and observed attributes equals to β, the reference monitor suffers at most and performs erroneous access decisions.

3) Flip Coin Enforcement: All formulas of a threshold enforcement suit for a flip coin enforcement too. Taking the assumptions done in the threshold enforcement and Definition 4, we obtain the average profit for a flip-coin enforcement per access request:

$$\langle C \rangle_{flip} = C_{tp} \cdot \beta^2 + (C_{fp} + C_{fn}) \cdot \beta \cdot (1 - \beta) \tag{11}$$

To find the minimal average cost, we should take the derivative of the average cost with respect to β. Hence the minimal average cost for the threshold enforcement is given by:

$$\beta_{min} = \frac{1}{2} \cdot \frac{C_{fp} + C_{fn}}{C_{fp} + C_{fn} - C_{tp}} \tag{12}$$

4) Example: We continue Example 1, and assume the security policy that grants access to "non-malicious" users. Let the initial attribute value for all access requests equals to "normal".

We impose the freshness uncertainty in the system (e.g., inevitable delays occurred during the attribute delivery and processing), that is, the reference monitor gets the attribute value $\tilde{A}_0 = 2$ at t_{permit} and can compute that there were exactly m attribute changes since t_{try} until t_{permit}. Then, we need to compute the probability β that the policy holds at t_{perm} and choose the model of the policy enforcement. We make an assumption, that the reference monitor knows the one-step transition matrix of the Markov chain. If so, the probability β is given by [10], [9], [8]:

$$\beta = \mathbf{Pr}[\mathcal{P}(A_m)|\tilde{A}_0 = 2] = \sum_{j \in \{1,2,3\}} (\mathrm{S} \cdot \mathbf{Prob}^{(m)})[j]$$

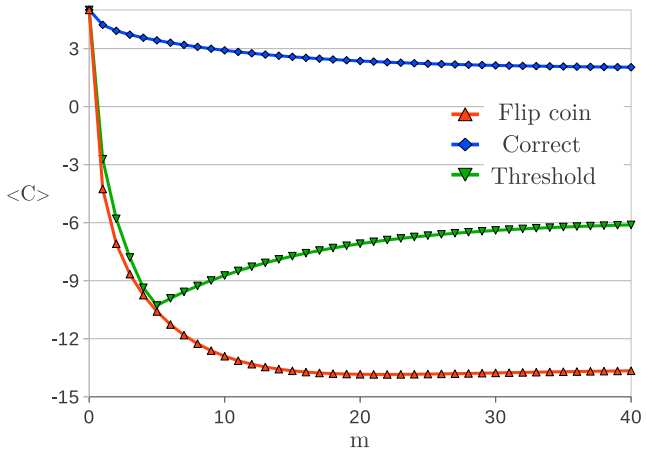

Fig. 3. Cost-Effective Enforcement of Access Control

where the vector S = [0; 1; 0; 0] specifies the initial attribute value.

Next, we picked the following costs: $C_{tp} = 5$, $C_{fn} = -45$, $C_{fp} = -15$, and to query an attribute we pay $C_a = -0.2$.

Then, we performed a set of simulations to show which of the enforcement models is the most cost-effective, i.e., the most profitable from the prospective of the reference monitor.

We computed the average profit per access request for the *correct enforcement* $\langle C \rangle_{cor}$, for the *threshold enforcement* $\langle C \rangle_{th}$, and for the *flip-coin* enforcement $\langle C \rangle_{flip}$. We varied the uncertainties between real and observed attributes by increasing the number m of attribute changes occurred between t_{try} and t_{perm}. We started from $m = 0$ and went up to 40 unobserved attribute changes.

Figure 3 shows the obtained results. Obviously, the average profit per access request for the *correct enforcement* is always higher, since the reference never makes erroneous access decisions there.

The average cost of the threshold enforcement decays when the number of attribute changes is below 5. The cost grows afterwards converging to the value -6 when the number of attribute changes is above 30. In fact, all the models converge to the certain cost value when m increases. This happens, because the Markov chain which models the attribute behaviour also converges to a stationary distribution. Hence, our analysis is appropriate mostly for $m < 20$.

The flip coin enforcement shows the worse results with respect to the threshold enforcement, despite the interval when these two models almost match. The flip coin enforcement also decreases initially and turns up slowly only when m is bigger then 22. By the way, when $m > 5$ there are chances for the flip coin enforcement that the eligible users may get the access, while for the threshold enforcement all the access requests are denied. In fact, the cost-effectiveness compromises the resource availability in some cases.

C. Cost of Usage Control Enforcement

Due to space limitations, we just give some general insights how to compute the average cost of the usage session. For usage control, all real and observed attributes should satisfy the policy, and indeed it is hard for the reference monitor to catch the attribute value which violates the policy. Instead, we compute policy violation and satisfaction times for each check. Indeed, a time interval consists of policy violation and policy satisfaction sub-intervals which substitute each other with a time passage.

The semantics of costs for usage control corresponds to "pay-per-time-of-usage" attributes, and specifies the exact benefits and losses the system gains in a unit of time for a given usage session. The system receives profit if a policy holds on a time interval and this revenue is proportional to the duration of the interval. In opposite, the system suffers losses during the policy violation time. Hence, only two costs for usage control are meaningful, and they are encoded by c_{tp} and c_{fn}, respectively.

Let π_k denote the policy satisfaction time in the k-th check and $\overline{\pi}_k$ encode the policy violation time. Obviously, $\pi_k + \overline{\pi}_k = t_e - t_b$ where t_e and t_b are time borders of the k-th check. Let H^k be a probability that a policy holds on the k-th check, and $\overline{H}^k$ states the opposite. Hence, the expected cost of a single check is given by:

$$c_k = \pi_k \cdot c_{tp} \cdot \mathbf{Pr}[H^k] + \mathbf{Pr}[\overline{H}^k] \cdot (c_{tp} \cdot \pi_k + c_{fn} \cdot \overline{\pi}_k) + C_a \tag{13}$$

The expected cost of a single check will converge to a particular value with the growth of the check interval. Thus, for our estimation let $c_k = c$ be a constant with respect to an order of a check. After k checks we will gain $k \cdot c$.

What we miss so far is a probability that there will be exactly k checks during the usage session, and k-th policy re-evaluation forces the access revocation. This probability is the reference monitor estimate of the "difference" between real and observed attributes for usage control.

Let $\Pr[\overline{G}_{ch}]$ specify the probability that the access is revoked after k-th policy re-evaluation and assume it also remains constant with respect to an order of a check. Therefore, the average cost of a single usage session is given by:

$$\langle C_U \rangle = c \cdot \sum_{i=0}^{\infty} i \cdot (\Pr[G_{ch}])^i \cdot \Pr[\overline{G}_{ch}]$$

i.e. we have a geometric distribution in number of checks, where a success means the access revocation.

VII. Related Work

Unintentional uncertainties related to freshness of attributes can be seen as particular cases of *timeliness* and *currency* factors from Bouzeghoub and Peralta [4]. Freshness of the first type relates to the problem of defining the frequency of updates (timeliness), while freshness of the second and third types is caused by natural delays in delivery of the authorization information (currency).

There are several related work on risk in access and usage control. Aziz et al. [3] assess policies considering different types of risk - operational, combinatorial and conflict of

interest. The approach is focused on reconfiguration of policy in a way to reduce its risk and save its strength. Han et al. [7] describe the approach to pre-evaluate security of policy using risk before enforcement. We don't consider composing of policies and assume that they are created in a secure way. Instead, our approach discusses peculiarities of collecting uncertain attribute values and problems connected with this issue.

Several approaches [16], [6], [12] use risk assessment to analyze cost of possible outcomes of access and employ a cost-benefit analysis to make an access decision. These methods consider a static decision making process while our approach analyzes the dynamic behavior of the system.

Few methods describe trustworthiness of policy arguments and update mechanisms. Skalka et al. [15] discussed the approach to evaluate credentials for distributed authorization with risk. Next to paying attention to trustworthiness of attributes our approach is also focused on their freshness.

The approach proposed in [11] empowers the UCON model with risk assessment. This paper describes an approach for selection of service providers (data consumer) in a service oriented architecture (SOA). The model of risk-aware usage policy enforcement is devoted to another problem: enforcement of policies by a resource provider rather by a requestor and making a rational decision about further accesses.

An examples of dealing with uncertain attribute values in UCON is given in [2]. Each remote attribute is associated with a security label which represents the trusted status of the attribute, and could change as the result of the attribute update. Since updates can run on a remote host, the behavior identifies whether the current value of the attribute is trusted within a specific platform. The model examines how to ensure the correct enforcement of the UCON policy particularly if the reference monitor is placed on the requestor's side.

VIII. Conclusions and Future Work

We introduced the model of the cost-effective enforcement of $UCON_A$ policies. We classified uncertainties real attributes and observed attributes used to produce access decisions. We defined correct, threshold, and flip-coin policy enforcement models. We analysed cost-effectiveness of these models for access control.

As drawbacks, we do accept the assumption that the uncertainty can be modelled with the probability of the policy violation and this probabilities are known. In fact, there are inevitable difficulties on determining probabilities, and on assigning the costs.

For a future work we are going to address these issues. Besides, $UCON_A$ policies also contain *actions*, e.g. attribute updates and obligations whose fulfilment can be uncertain too. We would like to capture these uncertainties and focus more on the continuous policy enforcement. Last but not least, we would like to take into account the diversity of users and instead of accepting an universe Markov chain for all users to implement a Markov decision process (MDP) adopting on-fly to a particular user. In fact, relevance of our approach to a partially-observable MDP is worth further investigations.

References

[1] M. Abadi. Logic in access control. In *LICS '03: Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science*, page 228, Washington, DC, USA, 2003. IEEE Computer Society.

[2] M. Alam, X. Zhang, M. Nauman, T. Ali, and J.-P. Seifert. Model-based behavioral attestation. In *SACMAT '08: Proceedings of the 13th ACM symposium on Access control models and technologies*, pages 175–184, New York, NY, USA, 2008. ACM.

[3] B. Aziz, S. N. Foley, J. Herbert, and G. Swart. Reconfiguring role based access control policies using risk semantics. *J. High Speed Networks*, 15(3):261–273, 2006.

[4] M. Bouzeghoub and V. Peralta. A framework for analysis of data freshness. In *Proceedings of the International Workshop on Information Quality in Information Systems*, pages 59–67, 2004.

[5] M. L. Damiani, E. Bertino, and C. Silvestri. Approach to supporting continuity of usage in location-based access control. In *FTDCS '08: Proceedings of the 2008 12th IEEE International Workshop on Future Trends of Distributed Computing Systems*, pages 199–205, Washington, DC, USA, 2008. IEEE Computer Society.

[6] N. N. Diep, L. X. Hung, Y. Zhung, S. Lee, Y.-K. Lee, and H. Lee. Enforcing access control using risk assessment. *European Conference on Universal Multiservice Networks*, 0:419–424, 2007.

[7] Y. Han, Y. Hori, and K. Sakurai. Security policy pre-evaluation towards risk analysis. In *ISA '08: Proceedings of the 2008 International Conference on Information Security and Assurance (isa 2008)*, pages 415–420, Washington, DC, USA, 2008. IEEE Computer Society.

[8] O. C. Ibe. *Fundamentals of Applied Probability and Random Processes*. Elsevier Academic Press, 2005.

[9] L. Krautsevich, A. Lazouski, F. Martinelli, and A. Yautsiukhin. Influence of attribute freshness on decision making in usage control. To appear in proceedings of 6th International Workshop on Security and Trust Management: STM'10, 2010.

[10] L. Krautsevich, A. Lazouski, F. Martinelli, and A. Yautsiukhin. Risk-aware usage decision making in highly dynamic systems. *International Conference on Internet Monitoring and Protection: ICIMP'10*, pages 29–34, 2010.

[11] L. Krautsevich, A. Lazouski, F. Martinelli, and A. Yautsiukhin. Risk-based usage control for service oriented architecture. In *proccedings of 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing: PDP'10*, pages 641–648. IEEE Computer Society, 2010.

[12] Y. Li, H. Sun, Z. Chen, J. Ren, and H. Luo. Using trust and risk in access control for grid environment. *International Conference on Security Technology*, 0:13–16, 2008.

[13] S. Osborn, R. Sandhu, and Q. Munawer. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Trans. Inf. Syst. Secur.*, 3(2):85–106, 2000.

[14] F. B. Schneider, K. Walsh, and E. G. Sirer. Nexus authorization logic (nal): Design rationale and applications. Technical report, Cornell Computing and Information Science Technical Report, 2009.

[15] C. Skalka, X. S. Wang, and P. Chapin. Risk management for distributed authorization. *J. Comput. Secur.*, 15(4):447–489, 2007.

[16] L. Zhang, A. Brodsky, and S. Jajodia. Toward information sharing: Benefit and risk access control (barac). In *POLICY '06: Proceedings of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 45–53, Washington, DC, USA, 2006. IEEE Computer Society.

[17] X. Zhang, F. Parisi-Presicce, R. Sandhu, and J. Park. Formal model and policy specification of usage control. *ACM Trans. Inf. Syst. Secur.*, 8(4):351–387, 2005.