Cost-effective Enforcement of UCON 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) security policies must be re-evaluated each time when attributes change. Catching timely all changes is a challenging issue. A reference monitor should be aware with certainty when to pull fresh attributes which reside outside of its control. Otherwise, some attribute changes would be missed, and worse, these unnoticed changes might violate security policies. Attributes mutability hardens the correct policy enforcement.**

This paper proposes a cost-effective enforcement of UCON policies. We assign monetary outcomes for granting and revoking access to legitimate users and those whose attributes violate security policies. Additionally, we place a price paid to obtain fresh attribute values. We introduce and compare a set of policy enforcement models in terms of cost-efficiency to handle attributes mutability. Besides mutability, we also take into account other factors that affect attributes trustworthiness.

***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 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 based on security attributes [13], [14], i.e. assertions done by trusted peers about subjects and objects participating in access and usage control.

The UCON model proposed by R. Sandhu et al. [18] encompasses access and usage control scenarios and operates with mutable attributes to specify and enforce security policies. The continuous policy enforcement poses new problems which have never been addressed in access control: *when* should a reference monitor query fresh attributes and perform re-evaluation of the access decision? How attribute mutability affects the policy enforcement?

Access decision checks should be triggered when attributes change. When all attributes are *local* and reside under control of the reference monitor, the system can follow some sort of a locking protocol during policy enforcement [14]. For instance, the system suspends ongoing usage sessions before updating attributes, and awaits until all updates are executed. Then, the system unlocks attributes and pushes new values to the reference monitor which re-evaluates security policies.

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.

Continuous pushing of fresh attributes can be implemented through subscription mechanisms[1]. Although, such approach might be very expensive for volatile attributes since a secure channel between the reference monitor and the attribute provider should be keeping alive. Instead, light alarms could be configured to notify the reference monitor when thresholds are crossed on key attributes. Unfortunately, this affects the privacy since security policies are disclosed to the attribute provider.

Pulling attributes might be privacy-preserving and less expensive. Basic approach imposes a *periodic* inquiring of all attribute repositories and, if attributes change then a policy re-evaluation is initiated [17]. 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].

Krautsevich et al. [10], [9] introduced the model of *aperiodic* attributes pulling. Authors assumed that a mutable attribute could be modelled as a stochastic process (e.g., a Markov chain) and the reference monitor knows the parameters of this process. Possessing the attribute value at a particular time, the reference monitor computes the probability that attribute might change since that time and a new value violates the policy. With a time passage this probability grows and when it overcomes the specified threshold, the reference monitor pulls a fresh value and triggers the policy re-evaluation. However, even if the fresh attribute value satisfies the policy, there is still no guarantee that the policy was holding in-between the checks.

Also, a system faultiness, delays occurred during 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 enforcement. The impact of uncertainties associated with

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

[1]www.oasis-open.org/committees/wsn/

attributes should be tolerated by the reference monitor.

This paper introduces the cost-effective enforcement of UCON policies. We assign monetary outcomes for granting and revoking access to legitimate users and those whose attributes violate security policies. Additionally, we place a price paid to obtain fresh attribute values noticing that frequent attribute queries are too expensive whereas rare checks lead to losses due to possible policy violations. We introduce and compare a set of policy enforcement models in terms of cost-efficiency to handle attributes mutability. Besides mutability, we also take into account other factors that affect attributes trustworthiness. The obtained results allow estimating the average time of the usage session and the expected profit preserving security.

The main contributions of this paper are:

- identifying and estimating the impact of all uncertainties associated with attributes acquisition;
- introducing models of correct and effective enforcement;
- introducing a cost model for a policy enforcement and attribute acquisition;
- identifying new attributes querying strategies needed to support cost-effective policy enforcement.

The paper is structured as follows. Section II points to the UCON model. Section III introduces attribute acquisition models and enlists all types of uncertainties associated with attributes. Sections IV and V present models of correct and effective policy enforcement, respectively. Section VI outlines a cost model and estimates an average profit and time of policy enforcement. Section VII summarizes related works. Section VIII concludes the paper.

II. USAGE CONTROL

Usage control (UCON) [18] demands for persistent control over resources. Continuity of control is a specific feature of UCON intended to operate in an inconstant context. This inconstancy is a result of the entire usage process or caused by other uncontrollable factors. The context is formed by mutable attributes of a requesting subject, an accessed object and execution environment.

UCON security policies restrict subject's behaviour and define which usages for the subject are permitted. UCON policy statements are built using authorizations (predicates over subject and object attributes), conditions (predicates over environmental attributes), and obligations (actions that must be performed along usage process).

For the sake of simplicity, we consider a security policy consists of authorization and/or conditions predicates only: p_{pre} and p_{on}. p_{pre} denotes predicates that should be satisfied before granting access to a resource (access control), while p_{on} denotes predicates that should hold during the access execution (usage control).

Authorization predicates are assumed to consist of a set of attribute clauses, built of one attribute variable and a threshold, in a conjunctive normal form. An attribute value is a variable a which might take a value from (in)finite domain of attribute values A. The attribute clause (predicate) is a logical function mapping the attribute value to either true or false.

III. ATTRIBUTE MODEL

A. Real and Observed Attributes

Security policies are based on *remote* attributes with *observable mutability*. *Remote* means that the attribute is managed by a remote attribute provider which is not under control of the reference monitor enforcing the policy. *Observable mutability* means that the reference monitor observes only partially how the attribute changes in time, and it cannot influence (or even block) the attribute modifications.

An attribute might change in discrete points of time and this process is modelled via a finite sequence

$$RealA = \{(a_i^{real}, t_i) | a_i^{real} \in A, t_i \in T, \forall i : t_i < t_{i+1}\}$$

where each element (a_i^{real}, t_i) specifies the attribute value a_i changed at time t_i. T is a countable infinite set of natural numbers which models time ticks. During time interval $[t_i : t_{i+1}]$ the attribute does not change, and the attribute value at t_{curr} equals a_i^{real}, where $t_i < t_{curr} < t_{i+1}$.

Only the attribute provider can see $RealA$, while the reference monitor operates with a finite sequence of observed attributes specified via

$$ObsA = \{(\langle a_j^{obs}, t_j \rangle, \tilde{t}_j) | a_j^{obs} \in A, t_j \in T, \forall j : t_j \leq \tilde{t}_j\}$$

where $\langle a_j^{obs}, t_j \rangle$ corresponds to elements (a_i^{real}, t_i) of $RealA$, i.e. $\forall j = i$, $\langle a_j^{obs}, t_j \rangle = (a_i^{real}, t_i)$. $\tilde{t}_j$ specifies the time point when the attribute was delivered to the reference monitor and used to evaluate the policy. Attribute delivery and access decision making are time-consuming operations, thus $\tilde{t}_j$ is usually bigger than t_j (time when attribute was issued). Notice, that the attribute provider and the reference monitor share the same trusted clocks which start to work at $t_{try} = 0$.

B. Basic Attribute Acquisition Models

In access and usage control two basic attribute acquisition models exist: *push* and *pull*.

Definition 1: ***Push Model*** defines a scenario when each new attribute value is pushed from the attribute provider to the reference monitor. Formally, this means that $ObsA$ and $RealA$ have the same number of elements. i.e.

$$|ObsA| = |RealA|$$
$$\forall i,\ (a_i^{real}, t_i) \in RealA\ \exists j = i,\ (\langle a_j^{obs}, t_j \rangle, \tilde{t}_j) \in ObsA$$

where $|S|$ specifies a number of elements in a sequence S.

Definition 2: ***Pull Model*** defines a scenario when the reference monitor queries the attribute provider to give the current attribute value.

There could be the case when the reference monitor queries too often (more frequently than the attribute changes), and as

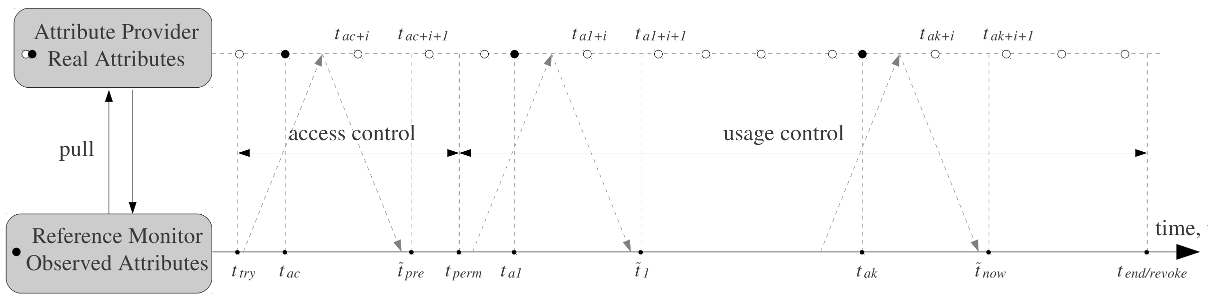

Fig. 1. Security Policy Enforcement with Pull Acquisition Model

a result $ObsA$ may contain **redundant** elements, i.e.

$$|ObsA| \geq |RealA|$$
$$RealA = \{ ..., (a_i^{real}, t_i), (a_{i+1}^{real}, t_{i+1}), ... \},$$
$$ObsA = \{ ..., (\langle a_j^{obs}, t_j \rangle, \tilde{t}_j), ..., (\langle a_{j+k}^{obs}, t_{j+k} \rangle, \tilde{t}_{j+k}), ... \},$$
$$\forall i, j\ \exists k \geq 0\ :\ t_i = t_j = ... = t_{j+k} < t_{i+1}$$

In opposite, the reference monitor might query too rarely (less frequently than the attribute changes), and as a result $ObsA$ may contain **insufficient** elements, i.e.

$$|ObsA| \leq |RealA|$$
$$RealA = \{ ..., (a_i^{real}, t_i), (a_{i+1}^{real}, t_{i+1}), ..., (a_{i+k}^{real}, t_{i+k}), ... \},$$
$$ObsA = \{ ..., (\langle a_j^{obs}, t_j \rangle, \tilde{t}_j), (\langle a_{j+1}^{obs}, t_{j+1} \rangle, \tilde{t}_{j+1}), ... \},$$
$$\forall i, j\ \exists k \geq 1\ :\ t_i = t_j,\ t_{i+k} = t_{j+1}$$

Obviously, the pull model with redundant elements is as expressive as the push model, i.e. each change of the attribute will be eventually captured by the reference monitor.

To the best of our knowledge, attributes pulling can be scheduled during usage control *periodically* and *aperiodically*.

C. Intentional and Unintentional Uncertainties

This subsection summarizes uncertainties associated with the attribute acquisition and discusses two types of uncertainties: *unintentional* which corresponds to a *freshness* and *correctness* of attributes, and *intentional* which corresponds to a *trustworthiness* of attributes.

1) Basic Notations to Measure Uncertainties: Let H_t be an event specifying that a real attribute value (a^{real}, t) does satisfy a security policy, i.e. $p_{pre/on}(a^{real}) = true$, while $\overline{H}_t$ specifies the opposite. The reference monitor operates only with observed attributes and can compute H_t^{obs}, i.e. the policy holds for observed attributes at a given time t.

Let G_t represent a fact that the reference monitor grants access (or continues a usage session) at time t. $\overline{G}_t$ states that the access is rejected (the usage session is revoked).

Usually, the reference monitor possesses *uncertain* knowledge about real attribute vales. Assume, the reference monitor can measure this uncertainty by computing the conditional probability $\mathbf{Pr}[H_t | H_t^{obs}]$ that the policy *really* holds at t knowing that observed attributes satisfy the policy at t.

Let $H_{[t_{perm}:\tilde{t}_{now}]}$ specifies that the sequence of real attribute values satisfies the policy p_{on} starting from t_{perm} till $\tilde{t}_{now}$.

In usage control the policy is evaluated every time when attribute changes. In case of the attribute pulling some values might be missed. Thus, the reference monitor has less information to prove that $H_{[t_{perm}:\tilde{t}_{now}]}$ holds. Assume, that knowledge of the reference monitor about the predicate satisfaction in this interval is probabilistic and

$$\mathbf{Pr}[H_{[t_{perm}:\tilde{t}_{now}]} | H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}]$$

specifies the probability that the policy really holds by $\tilde{t}_{now}$ knowing that observed attributes satisfy the policy at time of issuing.

2) Freshness of Attributes: is unintentional uncertainty occurring due to attributes mutability. Generally, it means that $H_t = H_t^{obs} \neq H_{t+\triangle t}$, i.e. attributes might change and new attribute value could violate the policy. We launched three types of freshness uncertainties.

Freshness I corresponds to the scenarios where only a part of attribute changes can be detected:

$$|ObsA| < |RealA|$$
$$\forall (\langle a_j^{obs}, t_j \rangle, \tilde{t}_j) \in ObsA,\ \exists (a_i^{real}, t_i) \in RealA$$
$$s.t.\ \langle a_j^{obs}, t_j \rangle = (a_i^{real}, t_i),\ and\ t_i = t_j = \tilde{t}_j$$
$$\mathbf{Pr}_{\mathbf{fr}}^{\mathbf{I}}[H_{[t_{perm}:\tilde{t}_{now}]} | H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}] < 1$$

As an example, assume the network of sensors provides 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 pulled attributes satisfy the policy.

Freshness II implies that an attribute may change during inevitable time delays $\triangle\ t_{proc} = \tilde{t} - t > 0$ needed for the delivery (due to a network latency) and decision making (evaluation of logical predicates). From the prospective of the reference monitor, freshness II means that

$$\forall (\langle a^{obs}, t \rangle, \tilde{t}) \in ObsA :\ t < \tilde{t}$$
$$for\ access\ control :\ \mathbf{Pr}_{\mathbf{fr}}^{\mathbf{II}}[H_{\tilde{t}} | H_t^{obs}] < 1$$
$$for\ usage\ control :\ \mathbf{Pr}_{\mathbf{fr}}^{\mathbf{II}}[H_{[t:\tilde{t}]} | H_t^{obs}] < 1$$

Freshness III corresponds to scenarios where the current attribute value is uncertain since some update queries are pending and may not be committed by the time of the policy re-evaluation.

In this case, the attribute provider sends two attributes: (i) the last certain attribute value and, (ii) *some additional information* on how the real value varies from the last certain:

$$(a_i^{real}, t_i), (\triangle\, a_i^{real}, t_{i+1}),\ where\, t_i < t_{i+1}$$

Accordingly, the reference monitor receives:

$$(\langle a_i^{real}, t_i\rangle, \tilde{t}_j), (\langle \triangle\, a_i^{real}, t_{i+1}\rangle, \tilde{t}_j),\ where\, t_{i+1} = \tilde{t}_j$$

As an example, assume a policy which allows users with a "normal" reputation to submit a huge number of applications for execution in Grid environment. 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 the 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 to be set up to make an access decision with some uncertainty on the current reputation of the user. For the given example, $\triangle\, a_i^{real}$ specifies how many applications were submitted for execution in interval $(t_i : t_{i+1})$ and are currently ongoing.

The presence of the uncertainty freshness III implies:

$$for\, access\, control:\ \mathbf{Pr}_{\mathbf{fr}}^{\mathbf{III}}[H_{\tilde{t_j}} | H_{t_i}^{obs}] < 1$$
$$for\, usage\, control:\ \mathbf{Pr}_{\mathbf{fr}}^{\mathbf{III}}[H_{[t_i:\tilde{t_j}]} | H_{t_i}^{obs}] < 1$$

Notice, the existing pull/push query standards (e.g. XACML, SAML) should be extended to support sending of two attributes upon the single request.

3) Correctness of Attributes: 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 attributes might differ from the real counterparts:

$$\forall(\langle a_j^{obs}, t_j\rangle, \tilde{t}_j) \in ObsA,\ \exists(a_i^{real}, t_i) \in RealA$$
$$s.t.\ a_j^{obs} \neq a_i^{real},\ and\, t_i = t_j = \tilde{t}_j,\ i.e.\ \triangle\, t_{proc} = 0$$

The presence of the correctness uncertainty implies that $\forall t, \mathbf{Pr}[H_t | H_t^{obs}] < 1$.

4) Trustworthiness of Attributes: 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 might respond with the same value which always satisfies the policy.

The presence of the trustworthiness uncertainty implies that $\forall t, \mathbf{Pr}[H_t | H_t^{obs}] < 1$.

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.

A. Correct Enforcement of Access Control

Figure 1 shows how the enforcement of access control evolves in time. It starts at t_{try}, and proceeds with querying current values of attributes. Either push or pull model is used, the reference monitor processes only the first received value and evaluates p_{pre} once.

Definition 3: ***(Correct Enforcement of Access Control)*** The reference monitor correctly grants access at $\tilde{t}_{pre}$ if the following holds (see Figure 1):

$$(p_{pre}(a_{ac+i}^{real}) = true) \wedge (p_{pre}(a_{ac}^{real}) = true)$$
$$RealA = \{\ (a_{try}^{real}, t_{try}),\ ...,\ (a_{ac}^{real}, t_{ac}),\ ..., (a_{ac+i}^{real}, t_{ac+i}),$$
$$(a_{ac+i+1}^{real}, t_{ac+i+1}) \ldots \}$$
$$ObsA = \{\ (\langle a_{ac}^{obs}, t_{ac}\rangle, \tilde{t}_{pre})\ \},\ t_{ac+i} \leq \tilde{t}_{pre} \leq t_{ac+i+1},\ i \geq 0$$

and the reference monitor is powerful to prove the following conditional probabilities:

$$\mathbf{Pr}[G_{\tilde{t}_{pre}} | H_{ac}] = \mathbf{Pr}[\overline{G}_{\tilde{t}_{pre}} | \overline{H}_{ac}] = 1$$
$$\mathbf{Pr}[\overline{G}_{\tilde{t}_{pre}} | H_{ac}] = \mathbf{Pr}[G_{\tilde{t}_{pre}} | \overline{H}_{ac}] = 0$$

where $H_{ac} = H_{t_{ac}} \cdot H_{\tilde{t}_{pre}}$, and it means that the reference monitor always grants the access when it should be granted, and denies otherwise.

Notice, the correct enforcement of access control requires that the predicates are satisfied exactly at t_{ac} and $\tilde{t}_{pre}$ although the policy might not hold in-between.

B. Correct Enforcement of Usage Control

Access control ends at t_{perm}, and usage control begins where the reference monitor re-evaluates p_{on} every time attributes change.

Definition 4: ***(Correct Enforcement of Usage Control)*** The reference monitor correctly continues the usage session by $\tilde{t}_{now}$ after evaluating the policy k times, if the following holds (see Figure 1):

$$p_{on}(a_m^{real}) = true,\ 1 \leq m \leq ak + i,$$
$$RealA = \{(a_1^{real}, t_1),\ ...,\ (a_{ak}^{real}, t_{ak}),\ ...,\ (a_{ak+i}^{real}, t_{ak+i}),\ \ldots \}$$
$$ObsA = \{\ ...,\ (\langle a_{ak}^{obs}, t_{ak}\rangle, \tilde{t}_{now})\}$$
$$t_{perm} = t_1,\ t_{ak+i} \leq \tilde{t}_{now} \leq t_{ak+i+1},\ i \geq 0$$

or the reference monitor revokes the access if the following holds:

$$\forall ak,\ s.t.\ (a_{ak}^{real}, t_{ak}) \in RealA,\ p_{on}(a_{ak}^{real}) = false,$$
$$\exists(\langle a_{ak}^{obs}, t_{ak}\rangle, \tilde{t}_{now}) \in ObsA,\ \tilde{t}_{now} = t_{ak}$$

and the reference monitor is powerful to prove that:

$$\mathbf{Pr}[G_{\tilde{t}_{now}}|H_{uc}] = \mathbf{Pr}[\overline{G}_{\tilde{t}_{now}}|\overline{H}_{uc}] = 1$$
$$\mathbf{Pr}[\overline{G}_{\tilde{t}_{now}}|H_{uc}] = \mathbf{Pr}[G_{\tilde{t}_{now}}|\overline{H}_{uc}] = 0$$

where $H_{uc} = H_{[t_{perm}:\tilde{t}_{now}]}$, and it means that the reference monitor continues the usage session when the policy holds, and revokes otherwise.

Proposition 1: The correct enforcement of access and usage control is possible if the attribute provider pushes certain attribute values and the processing time is negligible:

$$\forall j(\langle a_j^{obs}, t_j\rangle, \tilde{t}_j) \in ObsA,\ \tilde{t}_j = t_j,\ i.e.\ \triangle\, t_{proc} = 0$$

Proof: This comes from the definition.

V. Effective Policy Enforcement

Neither push no pull model can guarantee that observed attributes are equal to real in the presence of uncertainties. The correct policy enforcement implies that uncertainties associated with attribute are deterministic, while effective - probabilistic.

A. Effective Enforcement of Access Control

The reference monitor is powerful to compute:

$$\mathbf{Pr}_{RM} = \mathbf{Pr}[H_{ac}|H_{t_{ac}}^{obs}] = \mathbf{Pr}[H_{t_{ac}} \cdot H_{\tilde{t}_{pre}}|H_{t_{ac}}^{obs}]$$

or using conditional probabilities

$$\mathbf{Pr}_{RM} = \mathbf{Pr}_{\mathbf{cor*tr}}[H_{t_{ac}}|H_{t_{ac}}^{obs}] \cdot \mathbf{Pr}_{\mathbf{fr}}[H_{\tilde{t}_{pre}}|H_{t_{ac}}^{obs}] \quad (1)$$

where the first multiplier corresponds to trustworthiness and correctness of attributes, and the second - to freshness. Usually, all types of uncertainties exist in the system and summing these uncertainties implies the multiplication of their corresponding probabilities. How to calculate a probability of a freshness uncertainty is given in [10], [9].

*Definition 5: (**Effective Enforcement of Access Control**)* The reference monitor enforces the policy effectively by computing $\mathbf{Pr}_{RM}$ and picking one of the following strategies:

(i) *enforce by comparing with a threshold* $\mathbf{Pr}_{th}$: if $\mathbf{Pr}_{RM} \geq \mathbf{Pr}_{th}$, the reference monitor grants the access, and denies otherwise. Formally:

$$\mathbf{Pr}[G_{\tilde{t}_{pre}}|\mathbf{Pr}_{RM} \geq \mathbf{Pr}_{th}] = 1, \mathbf{Pr}[G_{\tilde{t}_{pre}}|\mathbf{Pr}_{RM} < \mathbf{Pr}_{th}] = 0$$

(ii) *enforce by flipping a coin* with $\mathbf{Pr}_{RM}$ of choosing a grant decision. If the coin flipped to grant, the reference monitor grants the access and denies otherwise. Formally:

$$\mathbf{Pr}[G_{\tilde{t}_{pre}}] = \mathbf{Pr}_{RM}$$

For better efficiency, we introduce a **non-oblivious** enforcement of a security policy. The basic idea is that the reference monitor grants the access and remains aware on additional information E suitable to estimate $\mathbf{Pr}_{RM}$ better. When this information arrives, the reference monitor recomputes previous positive access decisions. If the initial decision was erroneous, the reference monitor invokes some forensic security mechanisms and expects a reward for accesses which were granted wrongly. In some cases, the non-oblivious enforcement can be as cost-effective as the correct enforcement.

Let an attribute be uncertain because of freshness II. Assume that, the reference monitor grants the access and after some period of time queries the attribute value at $\tilde{t}_{pre}$. Let this value satisfy the policy, i.e. $E = H_{\tilde{t}_{pre}}$. Thus, the reference monitor recomputes $\mathbf{Pr}_{RM} = \mathbf{Pr}[H_{t_{ac}} \cdot H_{\tilde{t}_{pre}}|H_{t_{ac}} \cdot E] = 1$, and realises that the initial access was made *correctly*.

As another example, let the policy grant the access to the storage service if the user provides the self-signed assertion that information to store does not contain illegal data (i.e. circulation of it violates third party copyrights). The attribute uncertainty should be less then a specified threshold $\mathbf{Pr}_{th}$.

The attribute a is pushed with the initial request, and assume that only its trustworthiness is uncertain, i.e. $\mathbf{Pr}_{RM} = \mathbf{Pr}_{\mathbf{tr}}[H]$, where $H = H_{t_{ac}}|H_{t_{ac}}^{obs}$. If $\mathbf{Pr}_{RM} \geq \mathbf{Pr}_{th}$ holds, the reference monitor grants the access. Further, the system does not impose any control and the usage session can be ended only on the user's demand. Imagine, that during the access the reference monitor receives the evidence E from a trusted party, that the data stored by the user is illegal. In this case, the reference monitor understands that the initial access decision was incorrect. For the given example, assume that:

- $\mathbf{Pr}[E\,|\,H]$ denotes the probability of seeing the evidence E if the data is actually correct, i.e. the evidence is false positive;
- $\mathbf{Pr}[E\,|\,\overline{H}]$ denotes the probability of seeing the evidence E if the data is actually illegal. This probability always equals to 1;
- $\mathbf{Pr}[H \mid E]$ denotes the probability that the stored data is illegal if the evidence of such violation is present.

The reference monitor revises previous estimates of the attribute trustworthiness using *Bayesian inference* and the re-evaluated probability of the policy satisfaction is given by:

$$\mathbf{Pr}_{RM}^{new} = \mathbf{Pr}_{\mathbf{tr}}[H|E] =$$
$$= \frac{\mathbf{Pr}[E|H] \cdot \mathbf{Pr}_{\mathbf{tr}}[H]}{\mathbf{Pr}[E|H] \cdot \mathbf{Pr}_{\mathbf{tr}}[H] + \mathbf{Pr}[E|\overline{H}] \cdot \mathbf{Pr}_{\mathbf{tr}}[\overline{H}]}$$

In fact, if $\mathbf{Pr}_{RM}^{new} < \mathbf{Pr}_{th}$ the reference monitor revokes the access immediately upon receiving the evidence. Notice, that revision can be forward, and backward, i.e. the evidence may state that the stored content is perfectly good.

B. Effective Enforcement of Usage Control

Suppose, the reference monitor only pulls attributes, operates with observed attributes and evaluates the policy k times $H_{t_{a1}}^{obs}$, $H_{t_{a2}}^{obs}$, ..., $H_{t_{ak}}^{obs}$, then:

$$\mathbf{Pr}_{RM} = \mathbf{Pr}[H_{uc}|H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}] =$$
$$= \mathbf{Pr}[H_{[t_{perm}:t_{a1}]} \cdot H_{t_{a1}} \cdot ... \cdot H_{[t_{ak-1}:t_{ak}]} \cdot H_{t_{ak}} \cdot H_{[t_{ak}:\tilde{t}_{now}]}|$$
$$H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}]$$

Using conditional probabilities:

$$\mathbf{Pr}_{RM} = \mathbf{Pr}_{\mathbf{fr}}[H_{[t_{perm}:\tilde{t}_{now}]}|H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}] \cdot$$
$$\mathbf{Pr}_{\mathbf{cor*tr}}[H_{t_{a1}} \cdot H_{t_{a2}} \cdot ... \cdot H_{t_{ak}}|H_{t_{a1}}^{obs} \cdot H_{t_{a2}}^{obs} \cdot ... \cdot H_{t_{ak}}^{obs}]$$

The first probability corresponds to freshness of attributes, and the second to correctness and trustworthiness of attributes.

If trustworthiness and correctness remain constant, e.g. the attribute is measured always with the same precision, then:

$$\mathbf{Pr_{fr}}[H_{[t_{perm}:\tilde{t}_{now}]}|H^{obs}_{t_{a1}} \cdot \ldots \cdot H^{obs}_{t_{ak}}] \cdot (\mathbf{Pr_{cor*tr}}[H|H^{obs}])^k \quad (2)$$

Important result shown by this formula is that the probability of the policy satisfaction (in presence of all uncertainties) decays exponentially in number of attribute queries and policy checks.

Definition 6: ***(Effective Aperiodic Enforcement of Usage Control)*** The reference monitor effectively enforces the policy under uncertainties by time $\tilde{t}_{now}$ and:

1) computes the probability of the policy satisfaction since the last observed attribute:
$$\mathbf{Pr}_{RM} = \mathbf{Pr}[H_{[t_{ak}:\tilde{t}_{now}]}|H^{obs}_{t_{ak}}]$$
2) idles until $\mathbf{Pr}_{RM} \geq \mathbf{Pr}_{th}$ and then performs attributes pulling;
3) if pulled attributes satisfy the policy - continue the usage session and go to 1, otherwise - revoke the access.

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. Additionally, we place a price paid to obtain fresh attribute values. Notice, the frequent attribute queries are too expensive whereas rare checks lead to losses due to possible policy violations. We estimate the *average time of the usage session* (t_{av}) and the *expected profit* (C_{av}) preserving security.

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 uncertain attributes, the reference monitor is unable to infer accurately whether the policy holds, and, consequently, to choose a good alternative. There are four scenarios how the reference monitor acts processing uncertain attributes:

- *true positive*: grant access and the policy holds;
- *false negative*: grant access and the policy is violated;
- *false positive*: deny access and the policy holds;
- *true negative*: deny access and the 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, cost, the reference monitor loses/gains choosing this scenario.

Let C^{ac}_{tp} denotes the cost of the true positive scenario, when the reference monitor grants the access operating with observed attributes and the policy really holds for real attributes. C^{ac}_{fn}, C^{ac}_{fp}, C^{ac}_{tn} are costs of the remaining scenarios, respectively. The semantics of costs for access control corresponds to 'pay-per-access' attributes, and specifies the 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: $C^{ac}_{tp} \geq 0, C^{ac}_{tn} \geq 0$; and negative in the case of erroneous decisions: $C^{ac}_{fp} < 0, C^{ac}_{fn} < 0$. For simplicity, assume $C^{ac}_{tn} = 0$.

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. Costs for usage control are given by: c^{uc}_{tp}, c^{uc}_{fn}, c^{uc}_{fp}, c^{uc}_{tn}. Average cost of a usage session $C^{uc}_{av} = c^{uc} \cdot t_{av}$, where t_{av} is session's duration.

B. Costs of Attribute Queries

Let C^{ph} denotes a cost to push a current attribute value with a time stamp indicating when the last change happened. For pulling, costs are:

- C^{pl}_0 is paid to pull a current attribute value with a time stamp of the last change;
- C^{pl}_1 is paid to pull an attribute value at a given time t;
- C^{pl}_2 is paid to pull a time stamped attribute value and a number of changes since that;
- C^{pl}_3 is paid to pull all time stamped attribute changes during a time interval.

Obviously, $0 < C^{ph} = C^{pl}_0 \leq C^{pl}_1 < C^{pl}_2 < C^{pl}_3$

C. Cost of Access Control Enforcement

For access control, along with a cost of granting the access there is at least one cost paid to pull/push an attribute:

$$\begin{aligned} C^{ac}_{av} = C^{ac}_{tp} \cdot \mathbf{Pr}[G \cdot H] + C^{ac}_{fn} \cdot \mathbf{Pr}[G \cdot \overline{H}] \\ + C^{ac}_{fp} \cdot \mathbf{Pr}[\overline{G} \cdot H] + C^{ac}_{tn} \cdot \mathbf{Pr}[\overline{G} \cdot \overline{H}] + C^{ph} \end{aligned} \quad (3)$$

or using conditional probabilities:

$$\begin{aligned} C^{ac}_{av} = C^{ac}_{tp} \cdot \mathbf{Pr}[H] \cdot \mathbf{Pr}[G|H] + C^{ac}_{fn} \cdot \mathbf{Pr}[\overline{H}] \cdot \mathbf{Pr}[G|\overline{H}] \\ + C^{ac}_{fp} \cdot \mathbf{Pr}[H] \cdot \mathbf{Pr}[\overline{G}|H] + C^{ac}_{tn} \cdot \mathbf{Pr}[\overline{H}] \cdot \mathbf{Pr}[\overline{G}|\overline{H}] + C^{ph} \end{aligned}$$

1) Correct Enforcement: If the reference monitor enforces the policy correctly, then there are only 2 possible scenarios: true positive and true negative.

The average profit per access request is given by (see Equation 3):

$$C^{cor}_{av} = C^{ac}_{tp} \cdot \mathbf{Pr}[H_{ac}] + C^{ph}$$

2) Effective Enforcement: The reference monitor is powerful to compute the probability of the policy satisfaction operating with observed attributes. If the reference monitor enforces a policy *effectively by flipping a coin* it will gain the average profit per access request:

$$C^{flip}_{av} = C^{ac}_{tp} \cdot (\mathbf{Pr}[H_{ac}])^2 + (C^{ac}_{fn} + C^{ac}_{fp}) \cdot \mathbf{Pr}[\overline{H}_{ac}] \cdot \mathbf{Pr}[H_{ac}] + C^{ph}$$

When the reference monitor enforces a policy *effectively by comparing with a threshold* the average profit depends mostly on the selected $\mathbf{Pr}_{th}$. It can be inferred from [10], [9] that the maximum profit C_{av}^{thr} would be if:

$$\mathbf{Pr}_{th} = \frac{C_{tn}^{ac} - C_{fn}^{ac}}{C_{tp}^{ac} - C_{fn}^{ac} - C_{fp}^{ac} + C_{tn}^{ac}}$$

In case of *effective non-oblivious enforcement*, the average profit per access request is almost as in the case of the correct enforcement including the cost of additional queries during the access and possibility to deny the access to legitimate users:

$$C_{av}^{obl} = (C_{tp}^{ac} + C_1^{pl}) \cdot \mathbf{Pr}[H_{ac}] + C_{fp}^{ac} \cdot \mathbf{Pr}[\overline{G} \cdot H] + C^{ph}$$

3) Simulation Results: Following the approach in [10], [9] we consider a mutable attribute which encodes a reputation of a requester. The attribute mutability is modelled as a stochastic process, i.e. a time-homogeneous ergodic discrete Markov chain [8] (see Figure 2). The attribute domain is

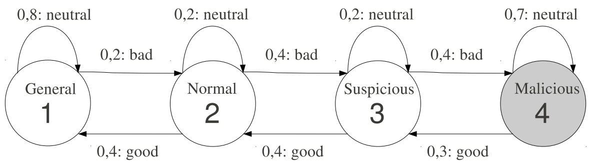

Fig. 2. A Reputation Attribute Model

$A = \{1, 2, 3, 4\}$ and let the policy deny the access if $a = 4$.

The attribute is pushed with the initial request at t_{ac}. Impose, that the value observed by the reference monitor is uncertain due to inevitable delays occurred during delivery (i.e. freshness II). Thus, the reference monitor needs to compute $\mathbf{Pr}_{RM}$ to enforce the policy effectively (see Equation 1). We make an assumption, that the reference monitor knows the one-step transition matrix of the Markov chain $\mathbf{Prob}$ (see Figure 2). Thus, $\mathbf{Pr}_{RM}$ is given by [10], [9]:

$$\mathbf{Pr}_{RM} = \mathbf{Pr}_{\mathbf{fr}}^{\mathbf{II}}[H_{\tilde{t}_{pre}} | H_{t_{ac}}^{obs}] = \sum_{j \in \{1,2,3\}} (\mathbf{S}_{t_{ac}}^T \cdot \mathbf{Prob}^n)[j]$$

where the vector $\mathbf{S}_{t_{ac}}$ specifies the probabilities distribution over good states at t_{ac}.

Let the attribute value a change every time tick, thus $n =^{\triangle} t_{proc} = \tilde{t}_{pre} - t_{ac}$ means that the attribute changed its value n times since the reference monitor observed it.

Next, we picked the following costs: $C_{tp}^{ac} = 50$, $C_{fn}^{ac} = -10$, $C_{fp}^{ac} = -5$, $C_{tn}^{ac} = 0$, and to query an attribute we pay $C^{ph} = -2$.

Then, we performed a set of simulations (100000 per each n) to show which of the enforcement models is the most cost-effective for a given access control scenario. We computed the average profit per access request for the *correct enforcement* C_{av}^{cor}, for the *effective by comparing with a threshold* C_{av}^{thr}, and for the *effective by flipping a coin* C_{av}^{flip}. We varied the time interval between the attribute was observed t_{ac} and the access decision enforced $\tilde{t}_{pre}$. In fact, as bigger this gap, the more changes of the attribute value occur.

Figure 3 shows the obtained results. Obviously, $\forall n : C_{av}^{cor} \geq C_{av}^{thr}, C_{av}^{cor} \geq C_{av}^{flip}$, i.e. the average profit per access request

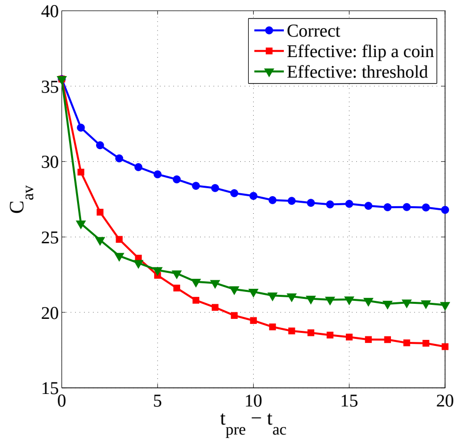

Fig. 3. Cost-effective Enforcement of Access Control

for the *correct enforcement* is always higher. Interestingly, that for $n \leq 4$ it is more profitable to enforce the policy flipping a coin, while for a big n comparing to the specified threshold is more effective.

D. Cost of Usage Control Enforcement

For usage control, we assume that a user never ends sessions on demand, and each session will be eventually revoked due to the policy violation.

1) Correct Enforcement: The policy is enforced correctly if all attribute changes satisfy the policy. Thus, for usage control there is a geometric distribution for the policy satisfaction and the average usage time is given by:

$$t_{av}^{uc} = \lambda \sum_{k=0}^{\infty} (k \cdot \prod_{0 \leq i \leq k} \mathbf{Pr}[H_{t_i}])$$

where k is a number of policy checks, and λ denotes the average number of attribute changes between adjacent checks.

The average profit per usage session:

$$C_{av}^{uc} = t_{av}^{uc} c_{tp}^{uc} + C^{ph}(1 + \sum_{k=0}^{\infty} (k \cdot \prod_{0 \leq i \leq k} \mathbf{Pr}[H_{t_i}]))$$

If $C^{ph} = 0$ we receive the maximum possible profit for a usage session when security is preserved.

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 policies. We specified uncertainties associated with security attributes and defined correct and effective policy enforcement models. We analysed the most cost-effective enforcement models for access control scenarios. We highlighted the idea of the non-oblivious enforcement for each access decision done using uncertain attributes. We pointed to a bunch of novel attribute pull queries which may help enforcing usage control policies.

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 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 adopting on-fly to a particular user.

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, M. Nakae, M. J. Covington, and R. Sandhu. Toward a usage-based security framework for collaborative computing systems. *ACM Trans. Inf. Syst. Secur.*, 2008.

[18] 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.