Available online at www.sciencedirect.com
SciVerse ScienceDirect
journal homepage: www.elsevier.com/locate/cose
Computers & Security

Usage control in SIP-based multimedia delivery☆

*Georgios Karopoulos**, *Paolo Mori*, *Fabio Martinelli*

Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Via G. Moruzzi 1, 56124 Pisa, Italy

ARTICLE INFO

Article history:
Received 22 April 2013
Received in revised form
16 August 2013
Accepted 17 September 2013

Keywords:
SIP
Security
Access control
Usage control
Multimedia security

ABSTRACT

The Session Initiation Protocol (SIP) is an application layer signaling protocol for the creation, modification and termination of multimedia sessions and VoIP calls with one or more participants. While SIP operates in highly dynamic environments, in the current version its authorization support is based on traditional access control models. The main problem these models face is that they were designed many years ago, and under some circumstances they tend to be inadequate in modern highly dynamic environments. Usage Control (UCON), instead, is a model that supports the same operations as traditional access control models do, but it further enhances them with novel ones. In previous work, an architecture supporting continuous authorizations in SIP, based on the UCON model, was presented. In this article, an authorization support implementing the whole UCON model, including authorizations, obligations and conditions, has been integrated in a SIP system. Moreover, a testbed has been set up to experimentally evaluate the performance of the proposed security mechanism.

1. Introduction

Nowadays, an emerging proliferation of multimedia applications is observed, and networks like 3G even define a separate subsystem for managing multimedia content delivery, namely IP Multimedia Subsystem (IMS) (3GPP, 2010). The main characteristic of IMS is that it is based on protocols with open specifications, like the Session Initiation Protocol (SIP) (Rosenberg et al., 2002) for managing multimedia sessions, and Diameter (Calhoun et al., 2003) for authentication, authorization and accounting. It is foreseen that the very same protocols will play a central role in Next Generation Networks (NGNs) managing multimedia content delivery over interconnected networks presenting a high degree of heterogeneity. This environment, however, will demand an appropriate access control model in order to handle its highly dynamic characteristics.

Traditional access control models like Mandatory Access Control, Discretionary Access Control and Role-Based Access Control exist in the literature for a long time. Each of these models is based on a different approach, but a common feature is that they all perform the authorization decisions at request time only, i.e., before a subject accesses an object. The Usage Control model (UCON) (Park and Sandhu, 2004), instead, besides supporting all the concepts present in the aforementioned models, introduces new features, the most notable of which are mutable attributes and continuous enforcement of the security policy for the whole lifetime of an access.

☆ This work was partially supported by EU FP7-ICT project NESSoS (Network of Excellence on Engineering Secure Future Internet Software Services and Systems) under the grant agreement n. 256980.

* *Corresponding author*. Tel.: +39 050 315 2069; fax: +39 050 315 2593.
E-mail addresses: georgios.karopoulos@iit.cnr.it, gkar@aegean.gr (G. Karopoulos), paolo.mori@iit.cnr.it (P. Mori), fabio.martinelli@iit.cnr.it (F. Martinelli).

0167-4048/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cose.2013.09.005

In a highly dynamic environment like NGNs it is very probable for attributes, such as user's reputation, paired with subjects, objects and environment to change their value, even during the course of a session (*mutable* attributes). Therefore, an authorization decision based on mutable attributes may not hold any more while the access is in progress, thus violating the security policy of the system. In these circumstances, traditional access control models are inadequate, whereas UCON supports the continuous re-evaluation of the security policy to interrupt unauthorized accesses while in progress.

In a previous work (Martini et al., 2011) the concept of ongoing authorizations on network resources was presented. We extended it in Karopoulos et al. (2012) with a more detailed presentation focused on SIP systems. In this article, we integrate an authorization system implementing the whole UCON model in a SIP system, with continuous enforcement of authorizations, obligations and conditions. We extended the prototype used in our previous work to support ongoing obligations as well, in order to experimentally evaluate the performance of our framework. The most important benefit of the proposed framework is the re-evaluation of the security policy during the exercise of access rights; this enhances system security avoiding the continuation of accesses when the corresponding rights expire.

The rest of the paper is organized as follows. Section 2 describes the motivation and the contribution of this paper. In Section 3 an overview of the UCON model is given, while in Section 4 the operation of SIP is briefly presented. The proposed method of continuous authorizations and obligations in SIP is analyzed in Section 5, followed by the description of our prototype and experimental results in Section 6. Related work is presented in Section 7 and Section 8 includes conclusions and future work.

2. Motivation and contribution

Multimedia sessions can be long lasting, i.e., many minutes or even hours (like traditional phone or video conference calls). However, even if the initial access to a SIP system has been authorized, some factors can change while the call is in progress in such a way that the corresponding access right does not hold any more. In this case, if traditional access control models are being used, the call will continue if none of the participants interrupts it, thus violating the security policy. For example, a free SIP system could require that an advertisement window is displayed on the caller's device while the call is in progress. Hence, if the caller closes this window while the call is still in progress, the security policy is violated. The UCON model can be adopted in SIP systems in order to regulate the usage of network resources in such cases. To address this issue, the UCON model enables us to define a security policy that must be satisfied for the whole duration of the SIP call. This means that the call is interrupted by the security enforcing mechanism as soon as this policy is not satisfied any more. For example, a predicate of the policy could state that the user reputation must be greater than T during the call. If the user reputation falls below the threshold T when the call is still in progress, the call is interrupted. Besides enhancing SIP systems security, this approach also allows to save network resources avoiding the continuation of unauthorized calls.

The main contribution of this article is the design of a complete framework implementing a UCON based authorization support for SIP systems, i.e., a framework supporting authorizations, obligations, conditions and continuous policy enforcement. It provides a detailed description of the authorization support architecture and its integration within the SIP system, focusing on the aspects concerning the implementation of the UCON model peculiarities, such as continuous enforcement of the policy and revocation of ongoing SIP sessions. Moreover, the article presents a complete set of experiments that evaluate the delay introduced by the proposed authorization mechanism.

3. Usage control

The UCON model, introduced in Park and Sandhu (2004), encompasses and extends traditional access control models introducing mutable attributes and new decision factors besides authorizations, i.e., obligations and conditions. *Mutable attributes* change their values as a consequence of the access decision process, and this could affect the same access or other accesses that are in progress. For instance, the value of the *reputation* attribute is decreased every time the subject tries to access an object but does not have the related rights. Traditional attributes (i.e., *immutable attributes*), instead, are modified only through administrative actions. For instance, the *role* attribute is updated when the subject gets a career advancement. Since mutable attributes can be updated during the usage of an object, in the following we show that each decision factor can be evaluated before (as in traditional models) and/or during the usage of the object (continuous control). Re-evaluating the access right when the access is in progress and interrupting this access when the related right is no more valid reduces the risk of misuse of resources.

Authorization predicates are evaluated to determine whether a subject requesting access to an object holds the corresponding right. This decision making phase takes into account subject/object attributes, and the action that the subject requested to perform on the object. The UCON model defines two categories of authorizations: pre-Authorizations (*preA*), where the decision phase is performed when the subject requests to access the object, and ongoing-Authorizations (*onA*), where the decision phase is performed while the access is in progress, in a continuous fashion.

Obligations are predicates that state whether certain requirements have been fulfilled in order to access objects. Pre-oBligation (*preB*) predicates verify whether some requirements have been fulfilled before the access, while ongoing-oBligations (*onB*) continuously check that the requirements are fulfilled while the access is in progress.

Conditions are requirements that do not depend on subjects or objects. They evaluate environmental or system status (e.g., current time or current location) to decide whether to allow access or not. A notable difference with respect to authorizations and obligations is that condition variables are not mutable and the evaluation of conditions cannot modify subject/object attributes.

4. The session initiation protocol

SIP (Rosenberg et al., 2002), is an application layer signaling protocol used in multimedia sessions and VoIP calls with one or more participants. SIP is used to set up a call only, and helps end users to negotiate the characteristics of the session.

In the classic operation flow, after the first part of the protocol where the caller locates the callee, the protocol is in essence a P2P one and users send SIP messages directly to each other. To terminate a call, any of the two users can send a BYE message directly to the other, utilizing SIP once again.

In the alternative operation flow, shown in Fig. 1, end users do not communicate directly with each other. The Back-to-Back User Agent (B2BUA) is a component of the SIP framework which operates between two communicating User Agents (UAs) and controls all signaling exchanged between them. It is not always present in a SIP architecture and this is the reason why it is not included in the classic SIP architecture. However, it is a basic element in the proposed scheme and this is why a brief overview of its operation is given here.

From the point of view of the calling SIP UA, a B2BUA acts as a user agent server (UAS) which receives requests and forwards them to the called SIP UA acting as a user agent client (UAC). This way, the two communicating end points never exchange SIP messages directly between them; if this element is present, even the call termination messages pass through the B2BUA as shown in Fig. 1. The benefits of using a B2BUA is that it can provide call management for the whole duration of a SIP dialog and full control over the calls.

5. Usage control in multimedia delivery

5.1. Architecture

Fig. 2 shows the general architecture of a system that provides multimedia services based on SIP together with an access control system based on the UCON model. The lines represent

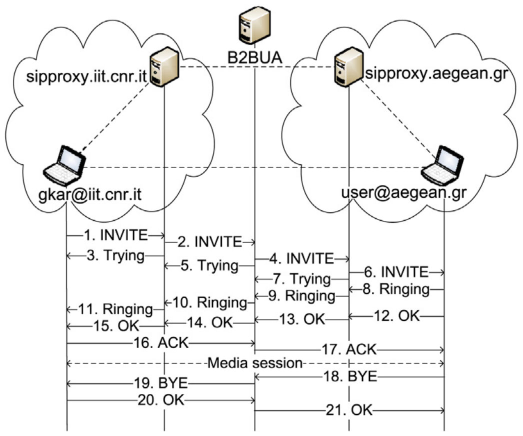

Fig. 1 – An alternative operation overview for SIP.

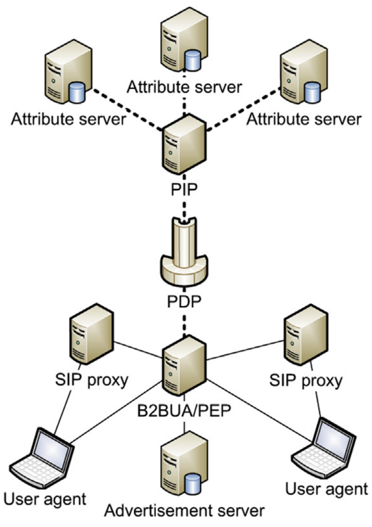

Fig. 2 – SIP access control architecture based on UCON.

the exchange of SIP messages while the dotted lines represent other protocols. In the proposed system all SIP signaling passes through a B2BUA in order to have full control over SIP sessions.

In the proposed architecture, the approach described in Yavatkar et al. (2000) is followed. According to this, there are two main entities: the Policy Decision Point (PDP) and the Policy Enforcement Point (PEP). The PDP is the component where the decision process is performed, and access decisions are taken, while the PEP has the responsibility to actually enforce these decisions by accepting or denying requests made by end users. Here an access control server, like a AAA server in de Laat et al. (2000), plays the role of a PDP; the PEP is co-located with the B2BUA which accepts session initiation requests from UAs.

The PDP contacts some attribute servers to retrieve updated values of subject/object attributes like user roles and user reputation. In particular, the PDP interacts with the Policy Information Point (PIP), that is the component of the authorization system that is in charge of interacting with the attribute servers, knowing their specific protocols.

When a SIP request arrives, the PDP invokes the PIP subscribing only for the attributes related to the user of this particular request. The PIP contacts the Attribute Servers to retrieve the current values of these attributes for a pre-evaluation of the security policy, and monitors these attributes to detect when their value changes, in order to trigger the PDP for a re-evaluation of the security policy. If the Attribute Server supports subscription, the PIP simply waits for a message from the Attribute Server. Instead, if the Attribute Server does not provide any subscription mechanism, the PIP periodically retrieves the updated values of user's attributes, and triggers the policy re-evaluation only if at least one of these is different from the previously collected ones. Attributes are updated as a consequence of the evaluation of the security policy that includes the update commands.

In other cases, specific obligations should be met before or during the provision of SIP services. As a representative example, in our architecture we suppose that obligations are realized through multimedia advertisements. In Fig. 2 the PDP is a UCON-aware server responsible for checking whether obligations should be met or not and a SIP Advertisement Server is utilized to deliver advertisement content to the user. When the B2BUA/PEP receives an "obligation needed" response from the PDP it creates a new SIP session between the caller and the multimedia server for delivering the advertisement.

5.2. *Security policy*

To express security policies we adopted the PolPA language. PolPA is a process algebra based language that allows to write history based security policies according to the UCON model (Martinelli and Mori, 2010; Martinelli et al., 2005). In particular, it exploits some composition operators to define the allowed behavior, i.e., *a*) the order in which security relevant actions can be performed, *b*) which authorizations, obligations and conditions must be satisfied to allow a given action, *c*) which authorizations, obligations and conditions must hold during the execution of actions, and *d*) which updates must be performed as a consequence of those actions. Roughly speaking, these operators allow to represent a sequence of actions, the alternative choice among a set of actions, the parallel execution of a set of actions, and the iterative or replicated execution of actions. For example, two or more actions must be executed in the same order as they appear in the policy if they are composed through a *seq* operator. Two or more actions can be executed alternatively or in parallel if they are related to an *or* or *par* composition operator, respectively. Moreover, PolPA allows to specify some predicates involving action's parameters and attributes of the user, the resource and the environment that need to be satisfied in order to proceed with the execution of the actions that follow the predicates in the policy. We use the commands *tryaccess(s, o, r)*, *permitaccess(s, o, r)*, *denyaccess(s, o, r)*, *endaccess(s, o, r)*, and *revokeaccess(s, o, r)* to represent the phases of an access, where *s* represents the name of the user that performs the action, *o* represents the name of the resource that is accessed, and *r* represents the specific security relevant action that implements the access along with its parameters. In particular, *tryaccess(s, o, r)* represents the request of the user *s* to perform an access, *permitaccess(s, o, r)/denyaccess(s, o, r)* represent the decision taken by the authorization system to allow/deny the access, and *revokeaccess(s, o, r)* represent the decision taken by the authorization system to interrupt an access that is in progress.

In the following sections we will show some examples for Authorizations and Obligations; we don't show any example for Conditions since both the policy and the message exchange is similar to the *pre-Authorization* case. For a detailed description of PolPA language refer to Martinelli and Mori (2010).

5.3. *Authorizations*

5.3.1. *Pre-authorizations*

In accordance to Section 3, Fig. 3 shows an example of *preA* in SIP, where the authorization procedure is executed as usual: any SIP call should be authorized before it is actually performed. Hence, the request of a UA for the creation of a new SIP session is authorized following the procedure in Fig. 3, and only if the PDP response is positive the two UAs start the exchange of multimedia data.

SIP session termination in *preA* scenarios is initiated when one of the two communicating edges sends a BYE message; this action triggers the procedure shown in Fig. 3 (messages 22–26). When the session is terminated the PEP embedded in the B2BUA informs the PDP (message 26).

A simple policy example implementing the *preA* model is shown in Table 1. This policy allows the execution of the call only if the reputation of the caller when the access request is performed is greater than a given value *R*.

The execution of a SIP call has been represented in PolPA using the security relevant action *sip_call*, whose parameters are the name of the receiver and the ID of the call, that starts when the SIP *INVITE* message is sent, and ends with the SIP

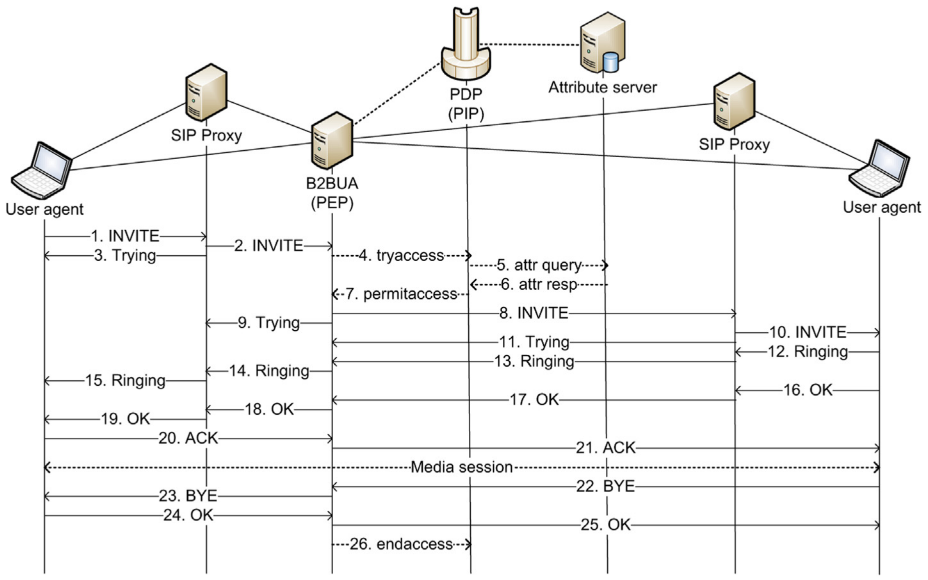

Fig. 3 – SIP pre-Authorization based on UCON.

Table 1 – Example of a *preA* security policy.

```
tryaccess(user, net, sip_call(recv,call_id)).           1
[(user.reputation > R)].                                2
permitaccess(user, net, sip_call(recv,call_id)).        3
(   endaccess(user, net, sip_call(recv,call_id))        4
or                                                      5
    endaccess(recv, net, sip_call(recv,call_id))        6
);                                                      7
```

BYE message. The PEP is in charge of intercepting these two SIP messages, sending to the PDP the request of the user to initiate a new call with the *tryaccess* command, and ending an existing call with the *endaccess* command. The first line of the policy represents the request of the user to initiate a call, and the permission is granted in line 3 only if the predicate in line 2 is satisfied. This predicate requires that the reputation of the user is greater than a given value *R*. In the *preA* model no further controls are executed while the call is in progress. The call can be terminated by any of the communicating users; line 4 and line 6 of the policy allow, respectively, either the caller or the receiver to terminate the call.

5.3.2. *Ongoing-authorizations*

In the *onA* model, the security policy is checked continuously while the media session is in progress and, as soon as any violation is observed, the authorization is revoked and the access is interrupted. Instead, if no violations occur, sessions are terminated by users, like in the *preA* scenario.

Fig. 4 shows a SIP session that is revoked during the media exchange. The PDP sends a subscription message to the attribute server (Fig. 4, message 5) to get the current values of the attributes related to the user that is initiating the SIP call, and to be notified when these values change. When the PDP is notified of an attribute update, the security policy is re-evaluated and if this results to a policy violation, an authorization revocations message is sent by the PDP to the B2BUA that closes the call (Fig. 4, message 23).

To revoke the dialog, B2BUA sends two BYE messages, one to each party and the dialog is terminated when they both respond with an OK message. Table 2 shows a policy example that extends the one in Table 1 by implementing the *onA* model. Again, here the execution of the call is allowed only if the reputation of the caller is greater that a given value *R* and the call is stopped as soon as the value of the caller's reputation is lower than *R* or one of the users terminate the call.

The first three lines represent the initiation of the call and are the same as in the *preA* example. After the call initiation there are two alternatives for terminating a call in progress: it can either be terminated by one of the users or be revoked by the PDP. Lines 4 and 6 of the policy represent the first case, in which one of the two users sends the termination command to close the call. In the second case, instead, the Attribute Server notifies the PDP about the updates of the attributes it subscribed, and the PDP, in turn, re-evaluates the policy. If the predicate in line 8 is satisfied, the PDP executes the *revokeaccess* command in line 9, that interrupts the execution of the call.

5.4. *Obligations*

5.4.1. *Pre-obligations*

Fig. 5 shows the sequence diagram in the case of *preB* model, i.e., when the execution of an obligation is a prerequisite for the establishment of a SIP session. The solid lines represent the exchange of SIP messages while the dotted lines represent other protocols; all SIP signaling passes through a B2BUA for having full control over SIP sessions.

In this example, the complete execution of an obligation is needed before the user can continue with the multimedia call. Fig. 5 presents an example where a multimedia advertisement has been chosen as the required obligation. Before the user's INVITE message is forwarded, a session with the advertisement server is established (message 5), and the user must watch or hear the advertisement; when the advertisement is over and the session has been terminated (message 11) the

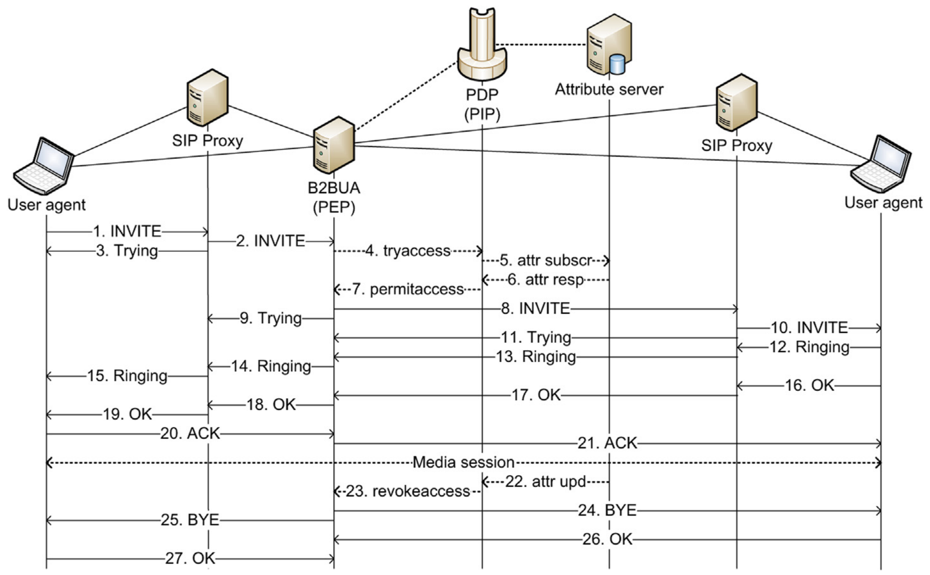

Fig. 4 – Authorization revocation while a SIP session is in progress.

Table 2 – Example of an *onA* security policy.

```
tryaccess(user, net, sip_call(recv,call_id)).                    1
[(user.reputation > R)].                                         2
permitaccess(user, net, sip_call(recv,call_id)).                 3
(   endaccess(user, net, sip_call(recv,call_id))                 4
or                                                               5
    endaccess(recv, net, sip_call(recv,call_id))                 6
or                                                               7
    ([(user.reputation ≤ R)].                                    8
    revokeaccess(user, net, sip_call(recv,call_id)))             9
);                                                               10
```

PEP notifies the PDP through the *endobl* message (message 12) that, in turn, answers sending the *permitaccess* message to the PEP (message 13). The PEP enforces the PDP decision forwarding the INVITE message to its destination (message 14).

Table 3 shows an example of *preB* model policy. This policy allows the execution of the call only if the user listens to or watches a given advertisement. As a matter of fact, the *tryaccess* in line 1 represents the user request to initiate a new call, that is authorized by the *permitaccess* in line 4 only if the execution of the obligation started by the *executeobl* command in line 2 is terminated, i.e., the *endobl* command has been received (line 3). The execution of the obligation is enforced by the PEP.

5.4.2. *Ongoing-obligations*

The sequence diagram in Fig. 6 shows the *onB* case, where the execution of an obligation takes place in parallel with the requested multimedia call. The user initiates a multimedia call by sending an INVITE message, the PEP requests access and the PDP responds back that the user should fulfil an ongoing-Obligation, such as watching an advertisement message. Then the B2BUA establishes a SIP session between the user and the advertisement server which should remain active for the whole period of the requested call between the two users. When the session between the two users is terminated, the advertisement session is also terminated. If the caller terminates the advertisement session while the call is still in progress, the B2BUA intercepts the termination and informs the PDP which requests the revocation of the call; this procedure is showed in Fig. 6, and the related policy is showed in Table 4.

The policy states that the execution of the obligation, represented by the *executeobl* command in line 2, starts as soon as the access request (i.e., the *tryaccess* command in line 1) is received, and the access is permitted right after, by the *permitaccess* command in line 3. While the call is in progress, if the user stops the execution of the obligation, the *endobl* message is sent to the PDP, and the policy interrupts the access by sending the *revokeaccess* command to the PEP (see lines 8 and 9).

6. Our prototype

In order to experimentally evaluate the performance of the proposed system an appropriate testbed has been set up exploiting both in-house developed components and modified versions of open source software. This section describes the architecture and the implementation of the testbed, and presents a set of experiments that have been performed to evaluate the delay introduced by the UCON system enforcing *preA*, *onA*, *preB* and *onB* policies.

6.1. *Testbed architecture*

The general architecture of the testbed is shown in Fig. 7. This testbed is based on the architecture presented in Section 5, and for each experiment we use the components we need. For the sake of simplicity, our testbed includes one administrative domain only, and thus one SIP server. Since a second SIP server would take part in non measured message exchanges, having one administrative domain will not affect our results. A brief analysis of the utilized software and hardware components follows.

6.1.1. *B2BUA*

Sippy (2011) is an open source SIP B2BUA server software based on Python. Our prototype exploits a modified version of Sippy embedding a PEP in order to contact the PDP every time

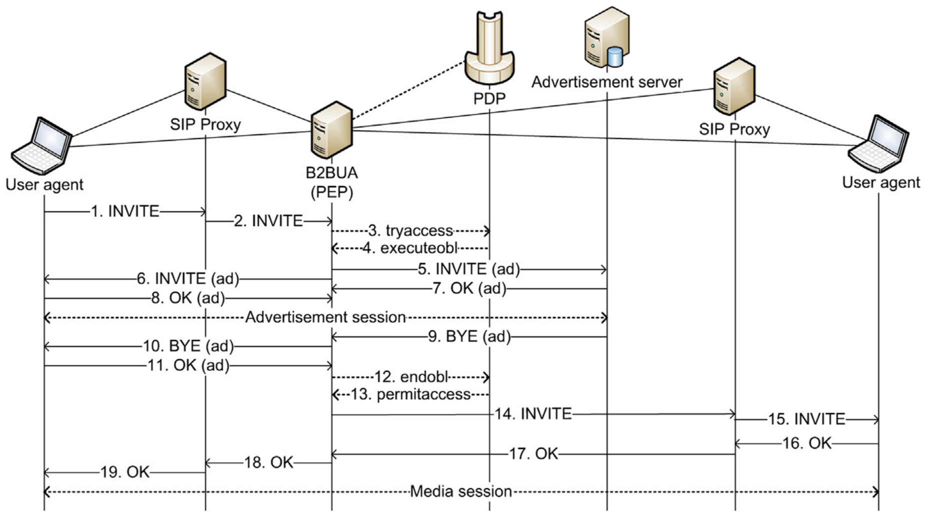

Fig. 5 – Executing an obligation before establishing a SIP session.

Table 3 – Example of a *preB* security policy.

```
tryaccess(user, net, sip_call(recv,call_id)).
executeobl(user, adv_id).
endobl(user, adv_id).
permitaccess(user, net, sip_call(recv,call_id)).
(   endaccess(user, net, sip_call(recv,call_id))
or
    endaccess(recv, net, sip_call(recv,call_id))
);
```

a user tries to perform a call. The PDP responds back either that the action is permitted or not, and could also ask the PEP to enforce obligations before and/or during the call. While the call is in progress the PEP waits for revocation commands from the PDP. In fact, PEP supports the revocation of ongoing sessions by sending BYE messages to the UAs involved in the session.

6.1.2. *SIP proxy server*

As SIP Proxy server, the open source SIP Router from the SIP Router Project (SIP Router, 2011) was utilized, which is a common development framework supported by similar but previously independent projects like: SIP Express Router (SER), Kamailio (OpenSER) and OpenIMSCore. They are all based on the SIP Express Router (SER) which is a high performance and configurable SIP server. In this case the configuration file of the server was modified in order to forward all SIP requests to Sippy.

6.1.3. *Policy decision point*

The PDP has been developed in IIT-CNR and supports usage control policies expressed using PolPA language (see Section 5.2). To implement the interactions between the PEP and the PDP a proprietary communication protocol was used.

6.1.4. *Attribute server*

In order to keep the architecture simple, we have chosen not to use a fully operational attribute server in our experiments. Instead, we use an ad-hoc component which modifies attribute values, thus triggering the authorization revocation procedures. However, this choice does not affect the time results since the measurement of the delays due to the authorization support starts when the attribute update message is sent to the PDP, i.e., it does not involve the Attribute Server.

6.1.5. *Advertisement server*

Our Advertisement Server is a SIP multimedia server which is responsible for presenting a multimedia advertisement to the caller during the *preB* and *onB* scenarios, and it is implemented as a SIP User Agent. Our experiments are focused on the SIP establishment and termination phases, thus the type of multimedia advertisement does not affect our results.

6.1.6. *User agents*

The User Agents used on the testbed are based on SIPp (2011) which is an open source test tool and traffic generator for SIP written in C++. SIPp operates according to scenarios defined in XML encoded files which describe simple as well as more complex call flows. Moreover, SIPp can play the role of a caller UA (client scenarios) as well as the one of the callee UA (server scenarios). For our testbed, we had to create appropriate client and server XML configuration files following the desired call flows. While SIPp is capable of producing different types of SIP messages, for our purposes we needed the generation of INVITE messages. Before, however, the proper routing of the messages to him, the intended recipient should register his current point of attachment before. This is accomplished by using sipsak (2006), a small open source command line tool used for simple tests on SIP applications and devices. With sipsak, in our testbed, an appropriate REGISTER message is send to the SIP Router so that it is aware of the point of attachment of the callee and able to forward the message.

6.1.7. *Machine specifications*

We run our experiments exploiting two machines on a local network. The first machine hosts the SIPp user agents

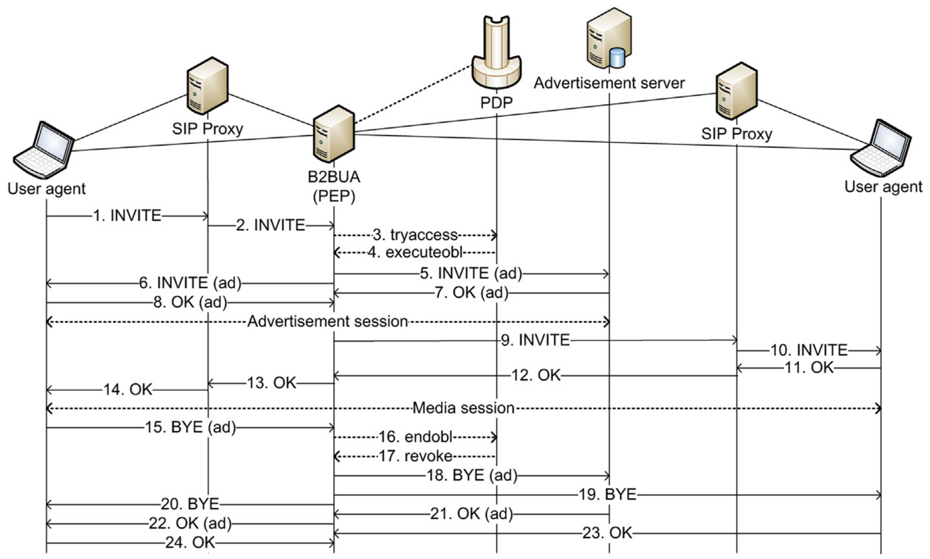

Fig. 6 – Terminating an obligation while a SIP session is in progress.

Table 4 – Example of an *onB* security policy.

```
tryaccess(user, net, sip_call(recv,call_id)).              1
executeobl(user, adv_id).                                  2
permitaccess(user, net, sip_call(recv,call_id)).           3
(   endaccess(user, net, sip_call(recv,call_id))           4
or                                                         5
    endaccess(recv, net, sip_call(recv,call_id))           6
or                                                         7
    (endobl(user, adv_id).                                 8
    revokeaccess(user, net, sip_call(recv,call_id)))       9
);                                                        10
```

including the Advertisement Server, the SIP router, Sippy and the scripts for attribute updates. The CPU is a Dual-core Intel Core 2 6600 running on 2.4 GHz, with 2 GB of memory, running a 32 bit openSuSE Linux version 11.3 with kernel version 2.6.34. In the second machine there is a VMWare player (2011) based Virtual Machine with the implementation of the PDP. The host machine's CPU is a Quad-core Intel Core i5 750 running on 2.67 GHz, with 8 GB of memory, running a 64 bit Ubuntu Linux version 10.10 with kernel version 2.6.35; the virtual machine is running a 32 bit Ubuntu Linux version 9.10 with kernel version 2.6.31 using 1 GB of memory.

6.2. *Experiments*

We have conducted two distinct sets of experiments for authorizations and obligations. Wireshark (2011) was used for intercepting all the messages exchanged among the components of our testbed.

6.2.1. *Authorizations*

The first set of experiments is aimed at measuring the delay introduced by authorizations. In particular, we measured *a*) the delay introduced by the *preA* phase, i.e., the delay for the initial authorization of a call, and *b*) the delay of the *onA* phase, which is the delay between the time when an attribute update that causes a security policy violation is issued, and the time that the respective session(s) is/are actually interrupted. In other words, this is the period that the call is still active despite the fact that a policy violation has occurred. Since the focus of these experiments was to measure the delay introduced by the authorization support only, we skip the authentication phase in the call set up. The following paragraphs describe the procedure we followed in order to get results for three distinct scenarios.

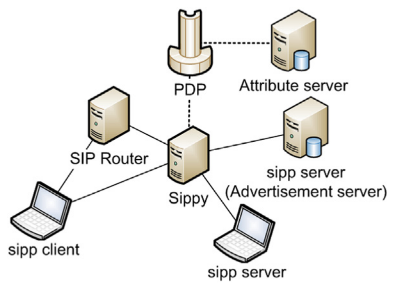

Fig. 7 – Our prototype architecture.

The experiments were conducted enforcing the simple policy shown in Table 2, that implements the *preA* model allowing the execution of the call only if the reputation of the caller is greater that a given value *R*, and implements also the *onA* model because the call is stopped as soon as the value of the caller's reputation is lower than *R*.

The first experiment measures the delay of authorizing the execution of a single SIP session, i.e., the time required for enforcing lines 1–3 of the security policy in Table 2. The results showed that, on average, the total time to set up a call, i.e. the interval from the time the caller initiates a call to the time the SIP device of the callee actually rings, including the authorization phase, is 19 ms. The average delay of the PDP to authorize the call, i.e. from the moment that the PEP sends the authorization request to the PDP until the PDP responds back, is 2 ms. Hence, the delay due to the authorization phase is 2 ms out of 19, i.e., about 10% of the total delay. This delay could increase if further authorization predicates are added in line 2 of the policy, but it does not depend on the number of active calls. In this case our results demonstrate that the delay imposed by the adoption of our UCON authorization system is so small that cannot be perceived by the communicating parties.

The second experiment measures the time required for the revocation of calls in progress; this corresponds to lines 8 and 9 of the security policy in Table 2. We suppose that a user has one active call and, while this call is in progress, his reputation falls below the minimum value allowed by the security policy and the PDP revokes the call. In order to measure the response times of the different components we have measured the delays of revoking one single session at a time from a queue of 1, 100, 200, ..., 1000 active calls. For each queue size the experiment has been repeated 100 times, and the average delay was calculated. The results of this experiment are shown in Fig. 8. The measured times are computed from the moment the attribute update message was sent to the PDP until the last *OK* message regarding the revoked session was received. Again, in this second experiment the results reveal that no significant delay is observed; in the worst case the total time required to stop a session is about 5 ms: less than 2 ms are due to the UCON authorization system, while the remaining time is due to the SIP system. Hence, this experiment shows that when

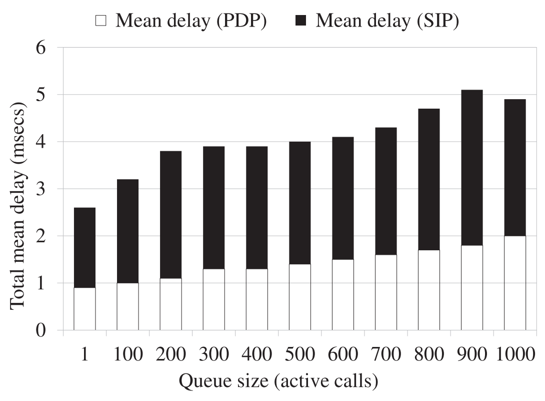

Fig. 8 – Mean delay of revoking an active SIP session with different active call queue sizes.

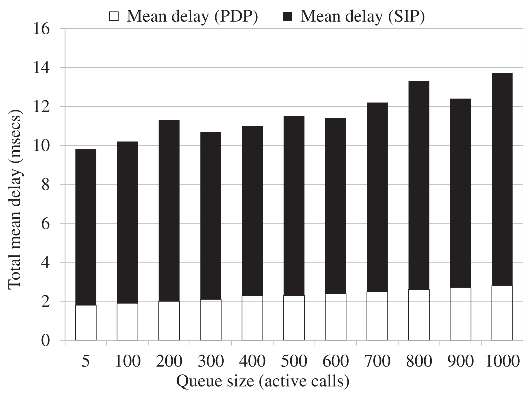

Fig. 9 – Mean delay of revoking 5 SIP sessions with different active call queue sizes.

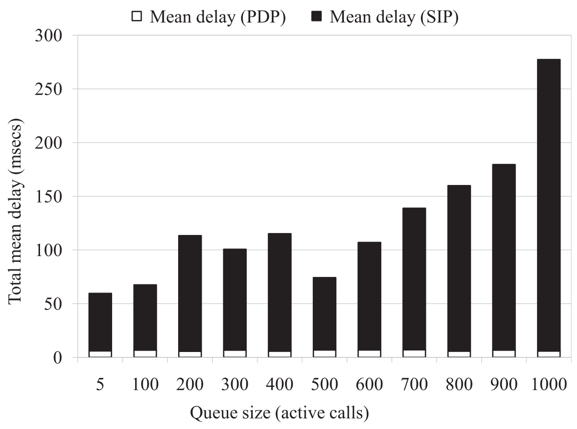

Fig. 11 – Mean delay of executing 5 obligations simultaneously with different active call queue sizes.

the right to execute a call expires, the call is interrupted very quickly.

The third experiment is similar to the second, but here the user has 5 active calls. In this case, 5 sessions are revoked and the delay is measured from the moment the attribute update message is sent to the PDP until the last *OK* message concerning the last of the five revoked sessions is received. Again, this was done 100 times for each size of the active calls queue, and the mean delay was calculated. The results of the third experiment are presented in Fig. 9. The figures show that the total time required to stop 5 calls reaches almost 14 ms in the worst case. However, the delay due to the UCON authorization system is a little more than 2 ms; hence, the largest part of the delay is due to SIP operations and it is not added by the proposed mechanism. To sum up, the experimental results in the case of authorizations demonstrate that the integration of the UCON authorization system within SIP architectures does not impose delays that are perceived by the end user, and allows to quickly interrupt calls when the corresponding rights have expired.

6.2.2. *Obligations*

The goal of the second set of experiments is to measure the delays introduced by the obligation management in the *preB* and *onB* scenarios. We have four scenarios in total, two for each case. Again in this set of experiments we skip the authentication phase during the call set up phase.

In the first scenario (*preB* case 1) we enforced the policy in Table 3 to measure the delay of executing an obligation for a single SIP session before the actual initiation of the session. In this scenario user A tries to call user B and the PDP, evaluating lines 1–3 of the policy, responds back that user A should watch an advertisement before his call is being set up. As a matter of fact, the *permitaccess* command (line 4), that authorizes the call set up, is placed after the *endobl* command (line 3), that states the natural termination of the execution of the obligation. Here we only measure the delay for the set up of the advertisement session leaving out the duration of the advertisement. In order to measure the response times of the different components we have measured the delays of setting up one advertisement session at a time when there are 1, 100, 200, ..., 1000 active user calls. The measured times are computed from the moment the first SIP INVITE message was sent from user A and includes the delays for setting up the session between user A and user B, the initiation and termination of the session between user A and the advertisement server, and the messages exchanged between the PEP and the

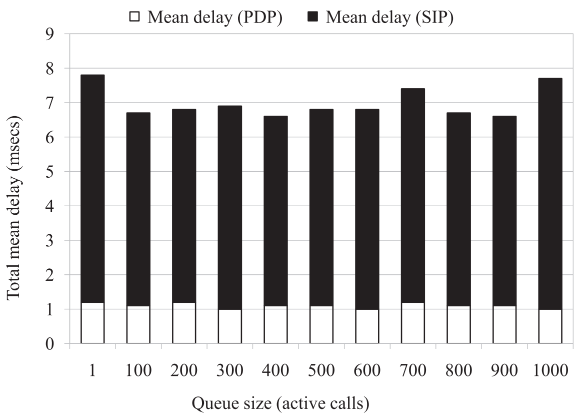

Fig. 10 – Mean delay of executing an obligation with different active call queue sizes.

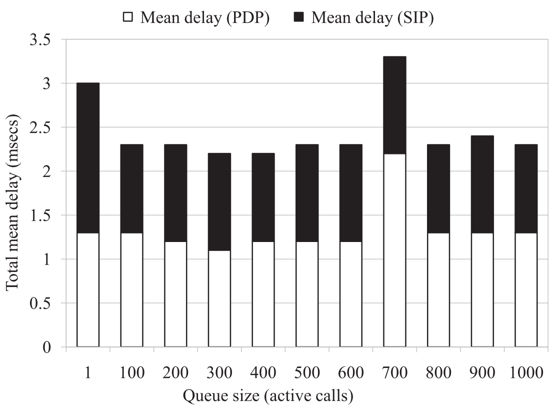

Fig. 12 – Mean delay of revoking a call after an obligation was terminated.

PDP. This experiment was performed 100 times and the mean delay was calculated. The results are shown in Fig. 10. In the worst case, the total delay for the *preB* scenario is near 8 ms out of which near 1 msec is due to the PDP operations. Hence, the delay introduced by the UCON authorization system is minimal in this case too.

The second scenario (*preB* case 2) is similar to the first one, but here user A initiates 5 calls at the same time, thus 5 advertisement sessions should be set up at the same time. The policy enforced is the one in Table 3. The measured times are computed from the moment the first call is initiated from user A until all advertisements have been set up and terminated and the 5th call has been accepted by the callee. Again, this experiment was done 100 times, i.e., 100 groups of five sessions were initiated one by one and the mean delay was calculated. The results of the second scenario are shown in Fig. 11. Here, a relatively higher total delay is observed which is around 280 ms in the worst case. However, the PDP delay is between 6 and 7 ms in all cases. Again, as in previous scenarios, the delay introduced by the UCON authorization system is low and the total delay is mainly due to SIP operations; one solution to this problem is the deployment of more than one SIP servers.

In the third scenario (*onB* case 1) we enforced the policy in Table 4. The user A initiates a call to user B; the PDP checks the security policy and responds that an advertisement video should be viewed in parallel with the call. The procedure is as following: user A initiates the call, the PEP requests a decision from the PDP, the advertisement session is being set up, and finally the call session is being set up. After a small period of time user A decides to terminate the advertisement; the PEP informs the PDP, which responds that the call session should be terminated. In this scenario we measure the delay from the moment user A terminates the advertisement session until both the advertisement and the call session have been terminated. In order to measure the response times of the different components we have measured the delays of terminating one single advertisement at a time when there are 1, 100, 200, ..., 1000 active calls. This experiment was repeated 100 times, thus, for each queue length, a single advertisement sessions was revoked 100 times and the mean delay was calculated. The results of the third scenario are shown in Fig. 12. In this scenario, the total time required to interrupt an ongoing call is very low because, in the worst case, we measured a delay of 3 ms, and the maximum delay due to the PDP is about 2 ms, like in the *onA* scenario. Hence, the time interval when the call is still active, although the corresponding right is not valid any longer, is negligible.

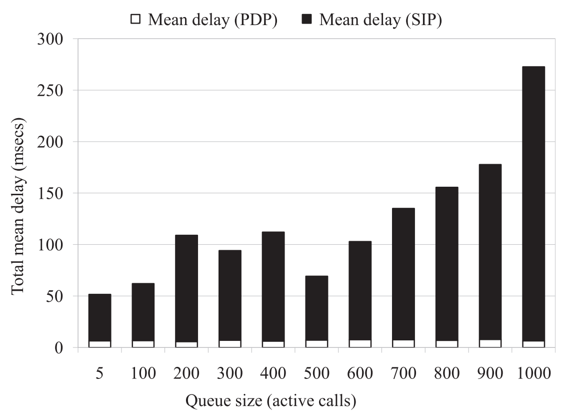

Fig. 13 – Mean delay of revoking 5 calls after 5 obligations were terminated simultaneously.

The fourth scenario (*onB* case 2) is similar to the third one. The only difference is that here user A initiates 5 calls, thus 5 advertisement sessions should be set up at the same time and be active in parallel with the call sessions. In this scenario, user A terminates these 5 advertisement sessions at the same time, so 5 call sessions should be revoked. The measured times are computed from the moment user A terminates the first of the 5 advertisement sessions until both the advertisement and the call sessions have been terminated. The average delay of 100 experiments was calculated. The results of this scenario are shown in Fig. 13. Here, a worst case total delay of 270 ms is observed, while the PDP delay is between 6.5 and 8 ms. Overall, the obligations experiments have shown that for 1 session the delays are minimal, both for the *preB* and for the *onB* scenario, i.e., the delay introduced to set up the call is not perceived by users (without the duration of the advertisement message) and unauthorized calls are interrupted very quickly. For 5 simultaneous sessions, instead, these delays increase significantly, but this is mainly due to SIP operations and they are not due to the policy evaluation performed by the PDP. Indeed, when revoking 5 authorizations, 20 SIP messages are exchanged through the PEP/Sippy; when revoking 5 obligations, 35 messages are exchanged. This causes the large difference between the first and the second case (Figs. 9 and 13 respectively) since the SIP server is overly stressed.

7. Related work

Previous approaches that incorporate the UCON model in SIP/IMS architectures are Martini et al. (2011); Karopoulos and Martinelli (2011); Karopoulos et al. (2012), and the work presented in this article builds upon them. In comparison to Martini et al. (2011), in the present work the usage control takes place on the SIP signaling level only and does not involve media exchange protocols. Karopoulos and Martinelli (2011) presents an approach for session management based on UCON for IMS and not purely for SIP. Karopoulos et al. (2012) concerns authorizations only; here, the whole UCON model for SIP is implemented.

SIP security is a well researched field covering a wide range of subjects. One domain covers SIP user identity security and issues related to it, like identity authentication (Rosenberg et al., 2002), privacy and anonymity (Peterson, 2002), and identity hiding (Karopoulos et al., 2011). The deployment of secure multimedia communications requires secure media transportation solutions as well. These solutions include the Secure Real-time Transport Protocol (SRTP) (Baugher et al., 2004) which is used in minisip (Minisip, 2013), and ZRTP (Zimmermann et al., 2011) which is used in Zfone (Zfone, 2013). A lot of research has been devoted to Denial of Service on SIP based services; an overview of issues and countermeasures is given in Sisalem et al. (2006). Another issue in SIP is SPIT which stands for SPAM over IP Telephony; Rosenberg and Jennings (2008) presents the problem and various possible solutions.

A SIP server is responsible for Authentication, Authorization, and Accounting (AAA). These procedures can be deployed in two different ways: *a*) embedded in the SIP server, and *b*) in an external server. In the second case, the most important protocols for the communication between the SIP server and the external AAA server are RADIUS (Rigney et al., 2000) and DIAMETER (Fajardo et al., 2012). RADIUS is older and more widely deployed than DIAMETER, while both of them are standardized by the IETF.

The UCON model has been adopted for designing the authorization system in several other scenarios.

In Zhang et al. (2006, 2008), the inventors of the UCON model describe how it can be adopted in collaborative computing systems, choosing the Grid environment as reference example. The proposed architecture is based on a centralized Attribute repository (AR) for attribute management. The values of the attributes are submitted to the authorization service by the user himself (push mode) for immutable attributes, while for mutable attributes the fresh values are collected by the authorization service just before their use (pull mode). The UCON policies are expressed using XACML.

In Martinelli and Mori (2010); Martinelli et al. (2005), instead, the authors define a process algebra based policy language, PolPA (the one adopted in this article), tailored for expressing UCON policies, and they show that all the UCON core models can be easily expressed. They also show an architecture for enforcing UCON policies in the Grid environment, to protect the providers of computational services from the applications they execute on behalf of Grid users. PolPA is also used in the mobile devices scenario (Costa et al., 2010). In particular, the proposed support performs a runtime monitoring of the operations performed by the Java applications executed on the mobile device.

In Stagni et al. (2009), the UCON model is adopted in Data Grid, i.e., Grid services that help users discover, transfer, and manipulate large datasets stored in distributed repositories and create and manage copies of these datasets. In this case too, authors used PolPA to express policies.

In Wang et al. (2006) the UCON model is adapted to protect services and devices in ubiquitous computing architectures.

Finally, Teigao et al. (2011); Xu et al. (2007) present an authorization system for an operating system kernel based on UCON. In this case, the UCON model is exploited to protect critical kernel resources such as kernel code, system call table, interrupt description table.

For further details about the UCON model and its applications in further scenarios, a general survey is presented in Lazouski et al. (2010).

8. Conclusions and future work

SIP security is a quite wide field and there are several aspects to be addressed, like confidentiality, integrity, availability; Section 7 offers insights and directions for finding solutions to these issues. Our proposal is focused on access control issues of SIP services and can be used in parallel and transparently with the aforementioned solutions in order to create a more secure SIP environment.

One of the main motivations that led to the proposal described in this article is that not much work concerning usage control on multimedia communications services has been done, despite the fact that Next Generation Networks is a highly dynamic environment where numerous users perform many interactions with/through the provided services. Since those interactions affect both users' and services' attribute values, the enforcement of access control policies should promptly react to these changes, in order to avoid the execution or continuation of unauthorized communications. In particular, the necessity of a new view on access control for multimedia services mainly stems from the fact that attributes, like user reputation or resource workload, constantly change with time leading to possible policy violations while a multimedia session is in progress.

For this reason, this article proposes the adoption of the UCON model, which can express traditional access control models as well as new concepts like attributes mutability, continuity of authorizations and enforcement of obligations. In order to validate the proposed approach, the article describes a possible architecture that integrates the UCON authorization system within the architecture of a multimedia service based on the SIP protocol, along with a set of examples of usage control policies that regulate the usage of this service. Moreover, in order to evaluate the impact on the performance of the multimedia services, this article also presents an implementation of the proposed system. The experimental results showed that our proposal is very efficient even with large volumes of active calls, and the revocation of active calls that break the security policy is done very quickly. Moreover, by observing the results it can be deduced that when multiple calls are revoked simultaneously, then the major part of the delay is rather due to SIP transactions than to PDP procedures, and this can be alleviated by using more than one SIP proxies per domain.

Hence, we conclude that the UCON model can be successfully adopted in the multimedia communication services scenario, because it enhances the service security without introducing a notable overhead in the system. Regarding the integration of the proposed Usage Control system within existing SIP systems, it requires a few modifications to the original architecture. If the B2BUA component is already installed in the SIP system, it must be updated to embed the PEP code; if not, then the B2BUA should be deployed and integrated by updating the configuration file of the SIP Proxy to forward all messages to it. Finally, the components of the Usage Control system, i.e., the PolPA PDP, the PIP and the Attribute server(s) should be deployed as well.

Our future work includes conducting more experiments with more complex security policies that are closer to real systems policies. Another aim is to conduct our experiments in more complex SIP architectures that employ more users and network elements; for example more SIP proxies could be used for the same or even for different administrative domains. As future work it is also worth considering the integration of UCON to other paradigms of multimedia communications, such as WebRTC. Last but not least, we plan to define a proper protocol (or to extend an existing one) for the interactions between the PDP and the PEPs to fully support the UCON model, which will also have some benefits such as supporting multi-domain environments.

REFERENCES

3GPP. TS 23.228: IP multimedia subsystem (IMS). Version 10.2.0, release 10. URL, http://www.3gpp.org/ftp/Specs/html-info/23228.htm; 2010.

Baugher M, McGrew D, Naslund M, Carrara E, Norrman K. The secure real-time transport protocol (SRTP); Mar. 2004. RFC 3711.

Calhoun P, Loughney J, Guttman E, Zorn G, Arkko J. Diameter base protocol. RFC 3588. Internet Engineering Task Force; Sep. 2003. URL, http://www.rfc-editor.org/rfc/rfc3588.txt.

Costa G, Martinelli F, Mori P, Schaefer C, Walter T. Runtime monitoring for next generation java me platform. Computers & Security 2010;29:74–87.

Fajardo V, Arkko J, Loughney J, Zorn G. Diameter base protocol. RFC 6733 (Proposed Standard). URL, http://www.ietf.org/rfc/rfc6733.txt; Oct. 2012.

Karopoulos G, Martinelli F. IMS session management based on usage control. In: Secure and trust computing, data management, and applications, 2011. STA 2011. The 8th FTRA international conference on. Springer; 2011.

Karopoulos G, Kambourakis G, Gritzalis S. PrivaSIP: ad-hoc identity privacy in SIP. Computer Standards & Interfaces 2011:301–14.

Karopoulos G, Mori P, Martinelli F. Continuous authorizations in sip with usage control. In: Euromicro conference on parallel, distributed, and network-based processing (PDP2012) 2012. p. 283–7.

de Laat C, Gross G, Gommans L, Vollbrecht J, Spence D. Generic AAA architecture. RFC 2903. Internet Engineering Task Force; Aug. 2000. URL, http://www.rfc-editor.org/rfc/rfc2903.txt.

Lazouski A, Martinelli F, Mori P. Usage control in computer security: a survey. Computer Science Review 2010;4(2):81–99. URL, http://www.sciencedirect.com/science/article/pii/S1574013710000146.

Martinelli F, Mori P. On usage control for grid systems. Future Generation Computer Systems 2010;26(7):1032–42. URL, http://www.sciencedirect.com/science/article/pii/S0167739X09001848.

Martinelli F, Mori P, Vaccarelli A. Towards continuous usage control on grid computational services. In: Joint international conference on autonomic and autonomous systems and international conference on networking and services, 2005(ICAS-ICNS 2005) oct. 2005. p. 82.

Martini B, Mori P, Martinelli F, Lazouski A, Castoldi P. Time-continuous authorization of network resources based on usage control. In: Trustworthy internet. p. 235–45. URL, http://link.springer.com/chapter/10.1007/978-88-470-1818-1_18; 2011.

Minisip. Minisip.org, open source secure high definition video conferencing. URL, http://minisip.org/; 2013. Last accessed on 28.02.13.

Park J, Sandhu R. The *UCON*$_{ABC}$ usage control model. ACM Transactions on Information System Security February 2004;7:128–74. http://doi.acm.org/10.1145/984334.984339.

Peterson J. A privacy mechanism for the session initiation protocol (SIP); Nov. 2002. RFC 3323.

Rigney C, Willens S, Rubens A, Simpson W. Remote authentication dial in user service (RADIUS). RFC 2865 (Draft Standard), updated by RFCs 2868, 3575, 5080. URL, http://www.ietf.org/rfc/rfc2865.txt; Jun. 2000.

Rosenberg J, Jennings C. The session initiation protocol (SIP) and spam. . URL. RFC 5039 (Informational), http://www.ietf.org/rfc/rfc5039.txt; Jan. 2008.

Rosenberg J, Schulzrinne H, Camarillo G, Johnston A, Peterson J, Sparks R, et al. SIP: session initiation protocol. RFC 3261 (Proposed Standard), updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621. URL, http://www.ietf.org/rfc/rfc3261.txt; June 2002.

SIP Router. The SIP router project. V3.1.1. URL, http://sip-router.org/; 2011. Last accessed on 28.02.13.

SIPp. SIPp: traffic generator for the SIP protocol. V3.2–TLS. URL, http://sipp.sourceforge.net/; 2011. Last accessed on 28.02.13.

Sippy. Session initiation protocol (SIP) back-to-back user agent (B2BUA) server software. V1.0.3. URL, http://www.b2bua.org/; 2011. Last accessed on 28.02.13.

sipsak. SIP Swiss army knife, a tool for simple tests on SIP. V0.9.6. URL, http://sipsak.org/; 2006. Last accessed on 28.02.13.

Sisalem D, Kuthan J, Ehlert S. Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms. Network, IEEE sept.-oct. 2006;20(5):26–31.

Stagni F, Arenas A, Aziz B, Martinelli F. On usage control in data grids. In: Ferrari E, Li N, Bertino E, Karabulut Y, editors. Trust management III. IFIP advances in information and communication technology, vol. 300. Springer Boston; 2009. p. 99–116.

Teigao R, Maziero C, Santin A. Applying a usage control model in an operating system kernel. Journal of Network and Computer Applications 2011;34(4):1342–52. Advanced Topics in Cloud Computing. URL, http://www.sciencedirect.com/science/article/pii/S1084804511000737.

VMWare player. VMWare player. V. 3.1.4. URL, http://www.vmware.com/products/player/; 2011. Last accessed on 28.02.13.

Wang H, Zhang Y, Cao J. Ubiquitous computing environments and its usage access control. In: Proceedings of the 1st international conference on scalable information systems. New York, NY, USA: ACM; 2006. InfoScale '06. URL, http://doi.acm.org/10.1145/1146847.1146853.

Wireshark. network protocol analyzer for Unix and Windows. V1.6.1. URL, http://www.wireshark.org/; 2011. Last accessed on 28.02.13.

Xu M, Jiang X, Sandhu R, Zhang X. Towards a VMM-based usage control framework for OS kernel integrity protection. In: Proceedings of the 12th ACM symposium on access control models and technologies. New York, NY, USA: ACM; 2007. p. 71–80. SACMAT '07. URL, http://doi.acm.org/10.1145/1266840.1266852.

Yavatkar R, Pendarakis DE, Guerin R. A framework for policy-based admission control. RFC 2753. Internet Engineering Task Force; Jan. 2000. URL, http://www.rfc-editor.org/rfc/rfc2753.txt.

Zfone. The Zfone project, secure VoIP phone software. URL, http://zfoneproject.com/; 2013.

Zhang X, Nakae M, Covington MJ, Sandhu R. A usage-based authorization framework for collaborative computing systems. In: Proceedings of the eleventh ACM symposium on access control models and technologies. New York, NY, USA: ACM; 2006. p. 180–9. SACMAT '06. URL, http://doi.acm.org/10.1145/1133058.1133084.

Zhang X, Nakae M, Covington MJ, Sandhu R. Toward a usage-based security framework for collaborative computing systems. ACM Transactions on Information System Security 2008;11(1):1–36. URL, http://doi.acm.org/10.1145/1330295.1330298.

Zimmermann P, Johnston A, Callas J. ZRTP: media path key agreement for unicast secure RTP. RFC 6189 (Informational). URL, http://www.ietf.org/rfc/rfc6189.txt; Apr. 2011.

Georgios Karopoulos received his M.Sc. in Information and Communication Systems Security in 2005, and his Ph.D. in Computer Network Security in 2009, both from the University of the Aegean, Greece. Currently he is a postdoctoral researcher at the

Information Security group of IIT-CNR, Italy. His research focuses in multimedia security and privacy, access and usage control, security in Next Generation Networks, and Critical Infrastructure protection. He is a member of the Technical Chamber of Greece.

Paolo Mori received his M.Sc. in Computer Science (cum laude) in 1998, and his Ph.D. in Computer Science in 2003, from the University of Pisa. He is currently a researcher of the Information Security group of IIT-CNR. His main research interests involve trust, security and privacy, focusing on access and usage control, trust and reputation management in distributed environments, such as Grid, Cloud, and mobile devices. His interests also include security in the Italian e-health scenario. He published 50 scientific papers and he has been/is involved in a number of European and Italian projects on information and communication security.

Fabio Martinelli received his Ph.D. in Computer Science from the University of Siena in 1999. He is a senior researcher of IIT-CNR. He is co-author of more than 100 scientific papers, and his research interests range from formal methods, distributed systems, computer security and foundations of security and trust. He founded and chaired the WG on security and trust management (STM) of the European Research Consortium in Informatics and Mathematics (ERCIM), and he is involved in several Steering Committees of international WGs and/or Conferences/ workshops.