2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing

# Self-Adaptive Authorization Framework for Policy Based RBAC/ABAC Models

Christopher Bailey, David W. Chadwick, Rogério de Lemos
School of Computing
University of Kent
Canterbury, UK
{c.bailey, d.w.chadwick, r.delemos}@kent.ac.uk

*Abstract*— **Authorization systems are an integral part of any network where resources need to be protected. They act as the gateway for providing (or denying) subjects (users) access to resources. As networks expand and organisations start to federate access to their resources, authorization infrastructures become increasingly difficult to manage. In this paper, we explore the potential of self-adaptive authorization as a means to automate the management of the access control configuration. We propose a Self-Adaptive Authorization Framework (SAAF) that is capable of managing any policy based distributed RBAC/ABAC authorization infrastructure. SAAF relies on a feedback control loop to monitor decisions (by policy decision points) of a target authorization infrastructure. These decisions are analysed to form a view of the subject's behaviour to decide whether to adapt the target authorization infrastructure. Adaptations are made in order to either endorse or restrict the identified behaviour, e.g. by loosening or tightening the current authorization policy. We demonstrate in terms of representative scenarios SAAF's ability for detecting abnormal behaviour, such as, misuse of access to system resources, proposing solutions that either prevent/endorse such behaviour, applying a cost function to each of these solutions, and executing the adaptive changes against a target authorization infrastructure.**

***Keywords*— *self-adaptation, authorization, autonomous access management, computing security, RBAC, ABAC***

## I. INTRODUCTION

A great deal of research and time is put into securing access to, and ensuring legitimate use of protected resources. There exist a variety of different approaches such as, role based access control (RBAC) [1] and attribute based access control (ABAC) [2], as well as more sophisticated systems involving detection [3], trust and feedback [4], and usage control [5] to compliment authorization. However, once authorization has been setup (i.e., defining authorisation policies) there exists few automated mechanisms that both identify when such access is being used incorrectly, and mitigate or prevent further misuse automatically. Traditionally organisations rely on audit trails and human administrators to monitor these systems to identify abnormal behaviour [6]. The detection of abnormal behaviour, attributed to the misuse of system resources by authorised subjects, is often not at the forefront of concern for organisations. However, it is known that an internally authorised user can cause far greater damage in comparison to an external attacker simply due to their access rights [7]. For example, during July 2010 it is alleged that a US army intelligence analyst downloaded millions of classified US military documents from a US Department of Defence website [8]. Assuming the analyst was an authorised user and that access was requested and granted on a document-by-document basis, we can say that the analyst had appropriate access rights and utilised the authorization system correctly. Any monitoring of the authorization system would not have picked up any abnormal behaviour as it processed the analyst's access requests according to its access control policies. However, to a human administrator numerous similar requests in a short period of time would have flagged up inappropriate behaviour, requiring immediate changes to the authorization infrastructure to mitigate any further damage.

Assuming that all subjects act appropriately within their access rights is an increasingly risky assumption to make, as organisations work together and federate their access control systems. As the number of subjects with federated access grows, resource holders are unaware of who is actually being granted access. It requires increased effort from system administrators to detect and prevent misuse. This position paper proposes that authorization infrastructures need to be capable of automatically identifying abnormal behaviour and autonomously change their configurations in order to either prevent further misuse or grant further access, in case of operational need. The contribution of this paper is the definition of a self-adaptive framework that manages a distributed RBAC/ABAC authorization infrastructure. We discuss behavioural analysis of subject interactions with the authorization infrastructure, alongside assessment of the impact of adaptations to enable effective 'self-adaptive' changes to the target authorization infrastructure. More specifically, we apply self-adaptive techniques in order to monitor, analyse, decide and manage a target authorization infrastructure.

The rest of this paper is structured as follows. In Section 2, we discuss related research. In Section 3, we describe our proposed framework. In Section 4, we outline the design of a pilot implementation of our framework in terms of scenarios. Finally Section 5 concludes with an evaluation of our proposal along with future work.

## II. RELATED WORK

This section discusses related research areas. We cover

RBAC/ABAC as our base technology, and look at complementary and alternative methodologies, before introducing self-adaptation.

## *A. RBAC / ABAC Authorization*

Role based access control (RBAC) and its more generic variant attribute based access control (ABAC) are models of authorization, facilitating access to protected resources through the use of roles/attributes and by assigning permissions to those roles/attributes. The RBAC/ABAC authorization model can be extended to include: hierarchy of roles/attributes, static separation of duties, and dynamic separation of duties. Our work is focused initially on core RBAC/ABAC over a distributed implementation.

A distributed RBAC/ABAC authorization infrastructure, as implemented in [9] comprises the following components:

- a set of distributed role/attribute issuing authorities, also known as Identity Providers (IdPs), which assign digitally signed credentials to subjects in a session,
- a Credential Validation Service (CVS) at the Service Provider's (SP) site, which validates the roles/attributes issued to the subject as credentials [10], and
- a Policy Decision Point (PDP) also at the SP's site, which evaluates if these roles/attributes give the user sufficient permission to access the requested resource.

Through the use of policies, attributes and credentials, subject authorization is provided. We refer to these as 'assets' of a distributed RBAC/ABAC authorization infrastructure. These assets demonstrate the parts of an authorization infrastructure that is changeable and therefore can be modified through self-adaptation to impact authorization decisions.

## *B. Usage Control*

Usage Control (UCON) [5] is an extension to the traditional access control model. The goal of UCON is to define rules that further control a user's access to protected resources, influencing access pre-, during and post-authorization. Assuming a user has the necessary access rights: obligations represent something a user must carry out before being authorised, and a condition is something that must be met before the user is given access. Mutable attributes represent 'changeable' attributes belonging to a user or the protected resource where access is required. Mutable attributes are treated as anything that can change as a result of access.

Paper [5] positions the idea that using these mutable attributes, in conjunction with obligations and conditions, policies can be defined in order to control when a user can be given access to a resource based on their usage. It also identifies the notion of continuity, where continuous enforcement takes place by evaluating usage rules through long-term sessions. The pretext to this could arguably be that usage policies are created in order to prevent misuse or abnormal behaviour. The limitations of this, however, are that usage control does not necessarily prevent further abnormal behaviour from continuing. For example, if a subject continuously breaks usage control rules no further action is taken (such as removing the subject's access rights), unless a human controller acts from reviewing these actions in audit logs. Furthermore usage control has no functionality to grant additional access right if the policy was originally too restrictive.

## *C. Intrusion Detection Systems*

Intrusion detection systems (IDS's) were introduced as a means of mitigating the risk involved with inherent flaws in security of computing systems. The concept of IDSs is well known, and the taxonomy of IDSs by Debar et al. [3] provides an overview of their properties. Primarily, IDSs are concerned with identifying and notifying administrators of potential attackers or potential past attacks, however, they can also be used to identifying legitimate user misuse. They aid an administrator's ability to monitor and detect attacks but rely on the administrator to act in light of attacks. IDSs detect intrusions using either knowledge-based or behaviour-based algorithms. Knowledge-based detection uses information about historical attacks to detect possible attacks. Behaviour-based uses a model of assumed normal behaviour comparing actions in a system by users to this model and identifying how far they deviate from that model.

IDSs can be beneficial for system administrators as they provide autonomous monitoring and detection, however they have certain limitations. In [3] an efficiency measure suggested is timeliness, whereby the IDS has a measure of the time required to analyse and report when an attack has happened or is happening. This could be considerably longer than the time required to grant or deny access requests, as the IDS has to work reactively, and is likely to require more processing time to determine if access requests are legitimate or not (due to more data processing about behaviour/knowledge). Subsequent to that, we must add the time it takes for an administrator to react to an alert in order to carry out preventative measures, which is highly dependent upon the administrator's availability and ability, resulting in a potential further loss/damage.

The above is known as passive detection. Some active intrusion detection systems do exist [3] whereby they react instead of waiting for an administrator. However, active IDSs create further limitations such as the accuracy of their decisions, false positives/negatives and gaining the trust of the human controllers.

## *D. Self-Adaptation*

Self-adaptation is a means of providing systems with the ability to adapt, manage, repair and update themselves automatically at run-time. This is often achieved through a feedback control loop [11] in which the system is monitored to obtain its current state, which is then compared to some previously planned or expected state in order to decide whether something needs to be adapted. There are various ways in which this can be achieved, which is highly dependent on the domain of the self-adaptive system and its purpose.

An example of a feedback loop applied to self-adaption of software systems is the MAPE-K (Monitor, Analyse, Plan, Effect – Knowledge) reference model [12]. This model identifies the core activities required for self-adapting a system. The role of the Monitor is to observe and record the state in the target system. The Analyser analyses the state for identifying the need for adaptation. The Planner generates plans based on the need for adaptation, and the Effector realises those plans in order for adaptation to happen. The knowledge part of MAPE-K is related to any information that enables the provision of self-adaptation, such as: models of the target system, goals that define what can be changed in a system [13], historical information about the use of the system, and previous successful or failed adaptive strategies.

## III. Self-Adaptive Authorization Framework

In this paper, we propose a Self-Adaptive Authorization Framework (SAAF) that is capable of being attached to any distributed policy based RBAC/ABAC authorization infrastructure. SAAF's objective is to autonomously monitor the usage of an authorization infrastructure, make judgements on the behaviour of subjects' interactions (in the form of authorisation requests and decisions), and adapt the target authorization infrastructure accordingly. SAAF is reactive, in the sense that it monitors the use of a target authorization infrastructure by subjects in order to detect abnormal behaviour. Once abnormal behaviour is detected a decision is made on whether to adapt the authorization infrastructure or not.

Conceptually, SAAF is based on the distributed RBAC/ABAC model shown in figure 1, and any adaption it carries out is bound to this. The conceptual model has 7 assets that are manageable. These are the attributes and credentials assigned to the subjects, the Attribute Authority's Credential Issuing Policy, and the resource owner's Credential Validation and Access Control Policies, collectively referred to as Authorization Policies (AZPs). The conceptual model provides a reference for SAAF to reason about the state of authorization.

Through changing the subject attributes we control what credentials may be issued, thus increasing or reducing the subject's permissions. The revocation of credentials allows for the termination of access sessions midway. Through the adaptation/switching of any of the

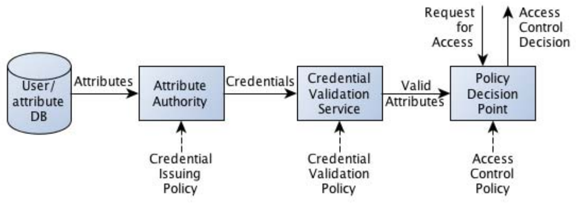


Figure 1. Conceptual Model for Distributed RBAC/ABAC Infrastructure

authorization policies, SAAF is able to impact a group of subjects by controlling authorization at a policy level. The authorization infrastructure interprets these assets in order to provide access control decisions. The modification of these assets by SAAF impacts the access control decision thus preventing or endorsing abnormal behaviour.

SAAF's operation relies on a collection of goals, a policy model containing behavioural rules and usage statistics. The *goals* represent what must be achieved through the management of the target authorization infrastructure, for example to minimise cost to the organisation.

The *policy model* portrays a generic view of the target's own authorization policies (AZPs), independently of the actual policy language used by the authorization infrastructure. Previous research has already defined a universal construct for RBAC/ABAC system policies [14], which is referred to as the ontology. The ontology provides the constructs of an RBAC/ABAC policy, and by populating those constructs we generate an abstraction of the given target's policies. Each policy within the *policy model* has an associated *meta–policy*. The meta-policy provides behavioural rules that are directly related to the authorization rules in the corresponding AZP, for example behavioural rules are associated to specific permissions. These rules represent statements of usage thresholds (upper and lower) that capture when the use of a permission becomes abnormal. Each behavioural rule is assigned a cost function denoting the impact to the organisation if the behavioural rule is broken. The policy model represents the current state of authorization. If SAAF needs to change that state (as a result of identified abnormal behaviour), the model is transformed to match those changes (i.e., modifying AZPs and removing subject attributes).

The *usage statistics* provide a historical view of the authorization infrastructure usage through monitoring of access requests and decisions being made. These allow statements of usage to be drawn about subjects, roles/attributes, and permissions for a certain period of time (e.g. average frequency of requests by role A or subject S to read resource R per minute during the last 30 days). This enables SAAF to identify how subjects are using the system collectively.

There are two categories of *cost* that are considered by SAAF: the cost of doing nothing and the cost of impact to the authorization infrastructure. The former must outweigh the latter for adaptation to take place. The latter refers to the impact that the adaptation might have on the organisation for example, the cost of removing a credential from a subject, or removing a role/attribute from a policy. The cost of executing adaptation is not considered.

In order for SAAF to be able to control a distributed RBAC/ABAC authorization infrastructure we make the following assumptions:

- the authorization infrastructure is capable of generating logs of its actions, e.g., failed and successful credential validations, failed and successful access requests, and that these logs are available to be read by SAAF;

- the authorization infrastructure has interfaces that allow it to receive new policies or replace old ones currently in use, and that SAAF can access these interfaces;
- identity providers are capable of allowing SAAF to modify the user attribute assignments, but if this is not possible then
- identity providers are capable of accepting notifications (from SAAF) about their user attribute mis-assignments and cases of abuse and are willing to remove and add new attributes to their users and notifies SAAF when the requested changes have been effected.

## *A. SAAF Architecture*

The architectural design of SAAF, shown in figure 2, embodies the MAPE-K reference model. The *monitor* is a simple component that retrieves assets of the authorisation infrastructure through observation. For example it captures an access request and corresponding authorization decision, sending it to the analyser. The *analyser's* objective is to process the monitored data in order to identify if abnormal behaviour has taken place. For this, it relies upon the policy model and historical usage obtained from the modeller. The analyser updates the modeller with analysed data (to be used for future analysis) and provides a set of solutions that may prevent or accept abnormal behaviour, to the planner. This is in the form of modifying rules belonging to an AZP and/or adding or removing a subject's attributes. The role of the *planner is to* select the most cost effective solution from the set of solutions provided by the analyser. The selected solution is then sent to the modeller for updating the policy model, and it is transformed into a plan that is sent to the effector. The *effector* adapts the authorization infrastructure in accordance to the plan, resulting in the modification of subject attributes/credentials and/or the management of authorization policies. The effector, along with the monitor, are the only system components that know which protocols,

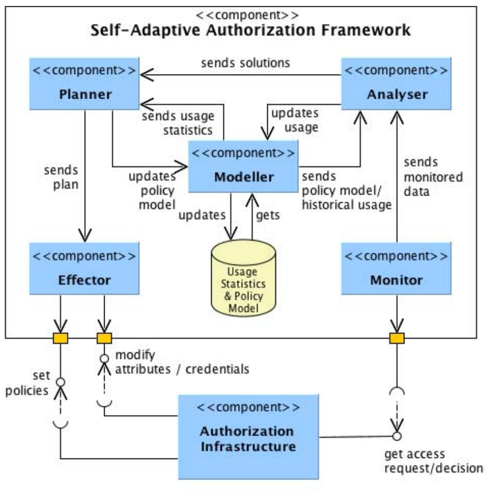


Figure 2. SAAF Architecture Diagram

interfaces and policy formats are needed for communicating with the various components of the target authorization infrastructure.

## *B. SAAF Adaptation Process*

The SAAF decision engine is the combination of the SAAF analyser and the SAAF planner. They perform four main activities: trigger adaptation, solution analysis, solution selection and plan generation. The engine's objective is to identify abnormal behaviour, in terms of misuse of access to resources, in order to provide a set of solutions to endorse or prevent such behaviour, select a solution, and realise that solution through adaptation. Although most adaptations serve to further restrict subjects' accesses to resources, there is one special case in which rules may be relaxed. This is so called "break the glass" rules that allow designated users to override deny decisions in exceptional circumstances [15]. If certain users are found to be continually breaking the glass, then SAAF may determine that the circumstance is no longer exceptional, and it manages policies/assigns attributes to grant these users normal access. The overall adaptation process is shown in figure 3.

### *1) Trigger Adaptation*

The trigger adaptation is SAAF's behavioural analyser, which identifies situations where an adaptation may need to occur. The monitored data that the analyser requires are the access request and corresponding decisions. It also requires the AZP from the Modeller.

Subject behaviour is assessed by analysing the subject's usage, in relation to the active AZP and their meta-policies (the policy model). The behavioural rules (stated within the AZP's meta policies) are restricted to what can be observed from a set of access requests. For example, from a set of subject requests we can identify a subject's active time, number of granted/denied requests made, and the rate of requests per time period. This allows us to assign behavioural rules about the frequency of requests for a given time period to specific permissions and attributes. This is akin to behavioural analysis performed in intrusion detection systems (IDSs) [3].

The trigger adaptation assesses if a subject or a role breaks any of the stated behaviour rules associated with a permission (or group of permissions) in the AZP. If a behaviour rule is broken, abnormal behaviour is detected, and solution analysis is carried out. Once the monitored data has been analysed it is used to update usage statistics via the modeller, for future reference.

### *2) Solution Analysis*

Solution analysis interprets the behaviour of subjects when utilising the authorization infrastructure based on the policy model. Solutions may exist in the form of alternative rules that are capable of preventing/endorsing the identified abnormal behaviour (e.g., role 'x' cannot read resource R). The analysis to be performed relies on the variables defined by the identified behaviour, for example, if a permission

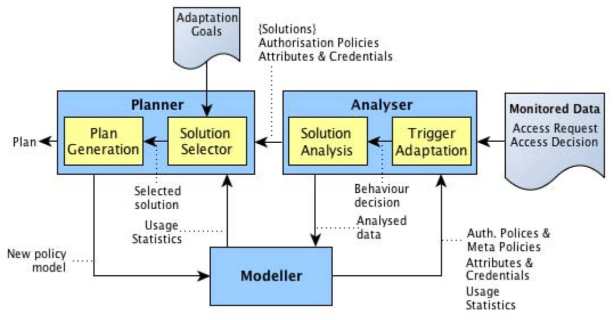


Figure 3. SAAF Adaptation Process

was misused by a subject (or role), the analysis would be focused around that subject, that role and that permission. The set of generated solutions is provided to the solution selector activity.

*3) Solution Selector*

The solution selector is part of the planner, and its role is to evaluate which solution should be transformed into a plan based on its adaptive goals (i.e. cost in this case). The adaptive goals dictate the selection criteria expressed as a multi-attribute decision problem. In this paper, goals are based on a single attribute, which is cost, e.g., the cost of removing someone's manager credential is £1000. The solution selector utilises a cost function to calculate the cost of each solution, and the cost of doing nothing. It then orders the solutions in terms of cost and discards those that are too expensive. For example, if a subject misuses their access rights to a printer, a solution may be to modify the policy where their role is no longer valid. However, the cost associated with this would mean that many subjects are impacted rather than just the offending subject. A less costly solution might be to request the identity provider (IdP) to remove the subject's role. There may be cases where all solutions are too costly, outweighing the cost of the current abnormal behaviour, and as a result, no adaptive change will occur.

*4) Plan Generation*

The generation of a plan identifies what actions need to be performed for realising the chosen solution. It can be viewed as an automatically generated set of step-by-step instructions with specific details on how to execute an adaptation [16]. For example, a plan may instruct the effector to "Remove rule A from the AZP and apply this to the AuthSys". As part of the generated plan, the planner updates the policy model in accordance to the required changes.

## IV. Pilot Implementation

This section considers the implementation of SAAF in the context of a library service that has a number of resources protected by an instance of the PERMIS authorization infrastructure [9]. Through the implementation, we demonstrate how SAAF can be integrated with PERMIS, and how it manages the authorization assets.

### *A. Implementation*

The library service relies on the PERMIS authorization infrastructure [9] that comprises an authorization server, and multiple Identity Providers (IdPs). The latter are trusted third parties that handle the authentication of subjects, and assign roles/attributes to those subjects as digitally signed credentials. PERMIS implements a distributed RBAC/ABAC model, relying on IdPs, CVSs and PDPs. Sources of Authority (SoAs) are the resource owners who provide the various policies to protect the resources and who say which IdPs are trusted to assign which attributes to whom.

The changeable assets of the system are the active AZPs, written as XML documents, and the subject's attributes, stored at the IdPs. Each AZP has a corresponding meta-policy provided for SAAF. The SoAs AZPs are written using PERMIS's own proprietary policy schema. These policies are stored as a digitally signed X.509 Attribute Certificates (ACs) for added security in the Library's own Lightweight Directory Access Protocol (LDAP) directory.

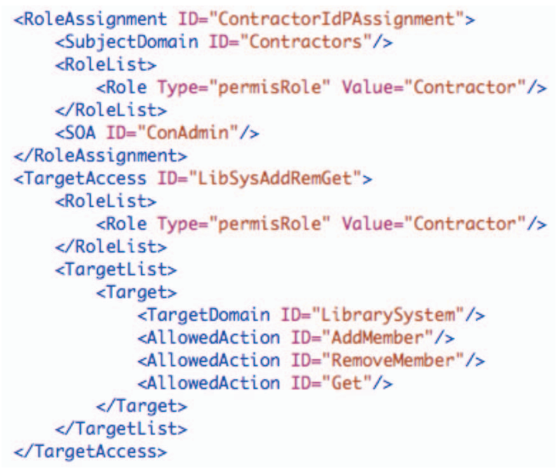
Attributes and credentials for subjects that require access to the library's resources are stored and assigned by two IdPs. One is the library's own IdP (LibAdmin) that assigns attributes for library members and staff, and the other is ConAdmin that assigns attributes for third party contractors. Each IdP stores their subjects' attributes in their own LDAP directory.

SAAF integration into PERMIS is achievable by automating the processes that are currently carried out by the library's SoA and IdP administrators. To do so, SAAF uses three interfaces, available from the authorization infrastructure, which allow it to: request/modify directly the subjects' attributes in each IdP's LDAP directory, and set the policies in the PERMIS authorization system. It is not able to modify the IdP's credential issuing policy, as this is hard coded into the software. Finally, it observes the access control requests and decisions processed by the PERMIS authorization system, which are recorded in logs maintained by PERMIS.

There are security concerns by allowing SAAF to control AZPs and LDAP attributes in the authorization system. Therefore, we utilise SSL/TLS to verify and authenticate SAAF, which the various components trust. We assume that SAAF is installed in the same administrative domain as the library system, and is provided with read access rights to the log file on the file system.

### *B. Adaptation Scenarios*

The library system has an electronic management system (LibrarySystem) protected by PERMIS. The current active AZP allows members of the role 'Staff' and 'Contractor' to access individual member details and add and remove them (see Figure 4). The AZP also allows members of the role



```
<RoleAssignment ID="ContractorIdPAssignment">
    <SubjectDomain ID="Contractors"/>
    <RoleList>
        <Role Type="permisRole" Value="Contractor"/>
    </RoleList>
    <SOA ID="ConAdmin"/>
</RoleAssignment>
<TargetAccess ID="LibSysAddRemGet">
    <RoleList>
        <Role Type="permisRole" Value="Contractor"/>
    </RoleList>
    <TargetList>
        <Target>
            <TargetDomain ID="LibrarySystem"/>
            <AllowedAction ID="AddMember"/>
            <AllowedAction ID="RemoveMember"/>
            <AllowedAction ID="Get"/>
        </Target>
    </TargetList>
</TargetAccess>
```

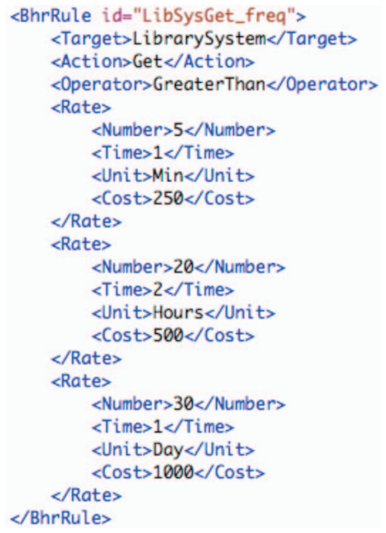
Figure 4. AZP Credential Validation Policy and Access Control Policy for Contractors

'Manager' to order new books, and members of the role 'Team Leader' to order new books if they first Break the Glass. The following scenarios demonstrate how access may be restricted or relaxed given certain behaviour captured by subjects of the library's protected resources.

*1) Restricting Access*

The library utilises an agency to supply contractors in place of full time staff when staff members are on leave. The library allows the agency to manage their subjects' access in terms of granting the agency's IdP 'ConAdmin' to assign subjects the role of 'Contractor'. The agency's IdP has been compromised by a group of hackers. They use the IdP's privileges to setup a number of fake bot subjects assigning them the role of 'Contractor'. The bots systematically use the library system's 'Get' action to steal details about library members.

The *trigger adaptation* reviews each of the bot's usage and assesses their current requests to the Get permission, as well as their historical usage (usage statistics) against the behavioural rules defined in the active AZP's meta-policy (see Figure 5). The behavioural rule states that the rate of requests for the Get permission must be no greater than 5 requests per minute, 20 requests per 2 hours, or 30 requests per day. The trigger adaptation remains in this cycle until abnormal behaviour is detected. If a subject's usage pattern breaks this rule the trigger adaptation activity identifies this as abnormal behaviour.



```
<BhrRule id="LibSysGet_freq">
    <Target>LibrarySystem</Target>
    <Action>Get</Action>
    <Operator>GreaterThan</Operator>
    <Rate>
        <Number>5</Number>
        <Time>1</Time>
        <Unit>Min</Unit>
        <Cost>250</Cost>
    </Rate>
    <Rate>
        <Number>20</Number>
        <Time>2</Time>
        <Unit>Hours</Unit>
        <Cost>500</Cost>
    </Rate>
    <Rate>
        <Number>30</Number>
        <Time>1</Time>
        <Unit>Day</Unit>
        <Cost>1000</Cost>
    </Rate>
</BhrRule>
```

Figure 5. AZP LibrarySystem.Get Behaviour Rule

Within seconds the bots' usage breaks the behaviour rule resulting in the trigger adaptation starting the solution analysis. *Solution analysis* assesses the broken rule (>5 accesses per minute), and identifies the 'Contractor' role and two bot subjects as the source of this behaviour. From this it generates the following solutions, as an attempt to remove the assets that permit the misbehaviour from occurring:

1. Revoke subject.bot1 & subject.bot2 active 'Contractor' credentials and alter credential issuing policy of ConAdmin so that 'Contractor' credential is no longer issued to subject.bot1 andsubject.bot2
2. Remove role 'Contractor' from subject.bot1 & subject.bot2 in ConAdmin's attribute database
3. Alter credential issuing policy of ConAdmin so that 'Contractor' credential is not issued to anyone
4. Alter credential validation policy so that 'Contractor' credentials from subject.bot1 and subject.bot2 are not trusted
5. Alter credential validation policy so that ConAdmin IdP is no longer trusted to issue 'Contractor' credentials
6. Alter access control policy to remove Get permissions from 'Contractor' credential
7. Remove 'Contractors' role from access control policy (and hence all its permissions)

Solution 1 removes the credential from the offending bots but does not remove their attributes, so they may still be able to use these for other actions. This is the minimum change possible to stop the offending behaviour, and is probably not enough, as other bots exist, which will continue to attack the library system. Solution 2 is more severe and removes the bot's attribute from the IdP's attribute database, therefore stopping them using the role of Contractor to access any resource anywhere. Solution 3 is yet more severe since it stops any subject from the IdP being assigned the 'Contractor' credential. However, in this case, this is probably the minimum change that will eventually be needed. Solutions 4 and 5 are similar to solutions 2 and 3 but alter the policy of the library system instead of the IdP. Solution 6 affects all contractors from all IdPs, since no Contractor can now execute the Get permission. Solution 7 is the most severe, since it removes all permissions from the 'Contractor' role issued by any IdP to anyone from anywhere. This is the change of last resort if no other change is effective. It is needed in cases where multiple IdPs are infected and are attacking the local site. It should only be carried out if the cost of doing so is less than the cost of the current attack.

The solutions are passed onto the *solution selector,* which computes the cost of each solution, as well as the cost of doing nothing (i.e. the current misbehaviour). All solutions, which are more costly than doing nothing, are discarded, the remainder are ordered in terms of cost. Solution 1 is initially chosen as the least costly one, and is sent to the planner. However, with continued abnormal behaviour being exhibited by all the bot subjects, SAAF will soon be triggered again and the solution selector eventually selects Solution 3 or 5, preventing the agency's IdP from assigning any credentials to its subjects for access to the library service.

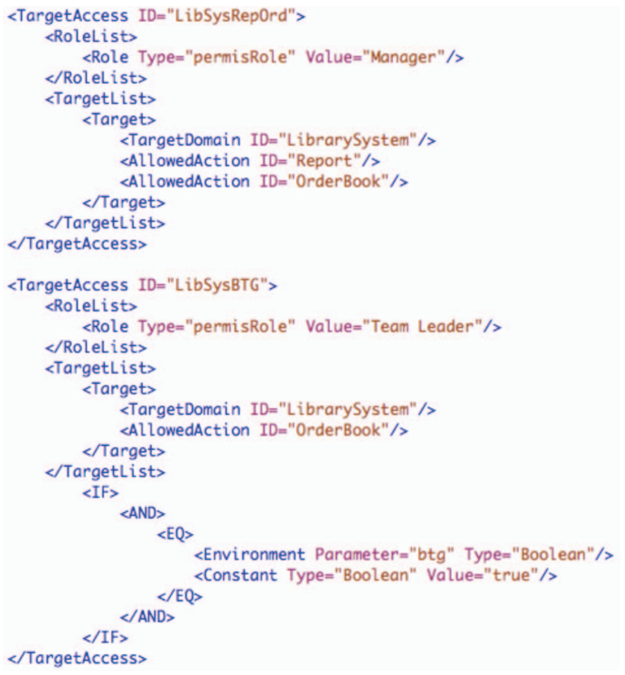
The *planner generation* initially forms a plan based on Solution 1 and passes it to the effector. The effector returns an exception, as it does not have the ability to either revoke short lived credentials or change the IdP's issuing policy. This causes the solution selector to opt for Solution 2, which is sent to the planner, then to the effector, which successfully removes the 'Contractor' attributes from the two bot's LDAP entries. Unfortunately this does not solve the problem, as other bots are continuing to attack the library system. Now the mis-behaviour is diagnosed as severe (<30 accesses per day) and Solution 5 is chosen. In this case, the plan provides a set of actions that result in the current credential validation policy being deactivated, and a new one, minus the rule that trusts the IdP ConAdmin, being activated. The effector executes this plan against the PERMIS authorisation system, invalidating all subjects' 'Contractor' credentials signed by the ConAdmin IdP.

*2) Relaxing Access*

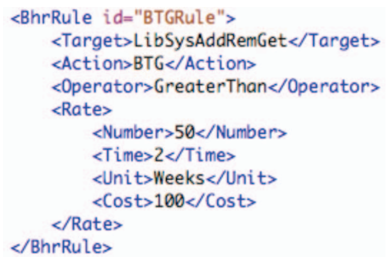
In this scenario, we show how an individual's usage results in the relaxing of access controls. PERMIS supports the use of break the glass (BTG) policies as described earlier. Abnormally high use of BTG permissions could suggest the need to relax the access control policy, as what were once considered emergency situations are now proving to be normal occurrences.

The library system has a subject 'Charlotte' who belongs to the role 'Manager'. She is capable of ordering new books and using the library system's reporting function (see figure 6). Mary is a subject with the role 'Team Leader' which grants Mary the right to order books if she is willing to break the glass and justify it. Recently, Charlotte has been very busy and has asked Mary to order many books for her.

SAAF analyses Mary's behaviour in relation to her BTG usage to decide whether an adaptation is necessary. As with the previous scenario, the *trigger adaptation* operates in a cycle analysing the requests made by Mary, assessing her actions (both BTG and book ordering) alongside historical usage statistics, and checking to see if she is breaking any behavioural rules. The trigger adaptation eventually identifies her usage breaks the BTG behaviour rule (greater than 50 BTG requests in 2 weeks) see figure 7. Note that Mary's book ordering behaviour is perfectly normal and


```
<TargetAccess ID="LibSysRepOrd">
    <RoleList>
        <Role Type="permisRole" Value="Manager"/>
    </RoleList>
    <TargetList>
        <Target>
            <TargetDomain ID="LibrarySystem"/>
            <AllowedAction ID="Report"/>
            <AllowedAction ID="OrderBook"/>
        </Target>
    </TargetList>
</TargetAccess>

<TargetAccess ID="LibSysBTG">
    <RoleList>
        <Role Type="permisRole" Value="Team Leader"/>
    </RoleList>
    <TargetList>
        <Target>
            <TargetDomain ID="LibrarySystem"/>
            <AllowedAction ID="OrderBook"/>
        </Target>
    </TargetList>
        <IF>
            <AND>
                <EQ>
                    <Environment Parameter="btg" Type="Boolean"/>
                    <Constant Type="Boolean" Value="true"/>
                </EQ>
            </AND>
        </IF>
</TargetAccess>
```
Figure 6. AZP LibrarySystem.Order permission and BTG permission


```
<BhrRule id="BTGRule">
    <Target>LibSysAddRemGet</Target>
    <Action>BTG</Action>
    <Operator>GreaterThan</Operator>
    <Rate>
        <Number>50</Number>
        <Time>2</Time>
        <Unit>Weeks</Unit>
        <Cost>100</Cost>
    </Rate>
</BhrRule>
```
Figure 7. AZP LibrarySystem.Order Behaviour Rule

does not break the behavioural rules, which are similar to those in figure 5.

*Solution analysis* identifies a set of solutions that endorse Mary's actions by identifying a means of increasing her access. For example, the analyser identifies what access Mary requires in terms of roles, permissions and attributes, and provides the following solutions:

1. Remove BTG condition from Team Leader role in access control policy LibrarySystem.OrderBook and add condition "if subject is Mary"
2. Remove BTG restriction from Team Leader role in access control policy LibrarySystem.OrderBook.

Solution 1 represents the need to update the current access control policy, allowing Mary to perform the action LibrarySystem.OrderBook. This is considered the safest and least costly solution, as it only impacts Mary. Solution 2 allows all Team Leaders to order books without breaking the glass. However this proposes more risk and is likely to be too costly, i.e., more expensive than doing nothing. The *solution selector* calculates the costs of both solutions' against doing nothing, resulting in the selection of Solution 1. As with the first scenario the planner will attempt to realise Solution 1 and form a plan, requesting the effector to execute the plan against the authorisation infrastructure.

## V. Conclusion

There is an inherent need for autonomic management of authorization infrastructures given the spread of protected resources and the existence of authorised users over multiple domains. In this paper, we have presented a Self-Adaptive Authorization Framework (SAAF) as a solution to autonomic management of authorization systems. The approach being proposed is focused on distributed RBAC/ABAC as an authorization model, and the MAPE-K autonomic computing reference model. We have described SAAF in an abstract way in order to promote its portability across different authorization infrastructures, and have designed a prototype for integration into the PERMIS authorization infrastructure. One advantage of SAAF, compared with more traditional approaches, is its responsiveness when reacting to circumstances that require the authorization system to adapt. SAAF has got some limitations, one being one that it requires a large amount of trust to be placed in it. SAAF must play the role of trusted ROOT and acts as the Source of Authority (SoA) for both the resource provider and the Identity Provider (IdP). Not all IdPs will be comfortable to allow a third party to affect their user attribute assignments, which is why we propose an alternative notification mechanism as well.

Our future work involves the further development of SAAF, specifically: the definition of cost functions, support for abnormal under use of resources, and more integration with risk management. We will draw upon work from trust access control [17], and cost associated trust access control [4], in order to build a formal framework for specifying clear controls that prevent wrongful adaptation. For the implementation of SAAF, the intent is to rely on model driven engineering alongside self-adaptation as a potential means of autonomously managing authorization systems.

## Acknowledgment

Co-financed by the Foundation for Science and Technology via project CMU-PT/ELE/0030/2009 and by FEDER via the «Programa Operacional Factores de Competitividade» of QREN with COMPETE reference: FCOMP-01-0124-FEDER-012983 and an EPSRC grant for studentship.

## References

[1] ANSI. "Information technology – Role Based Access Control". ANSI INCITS 359-2004.

[2] ITU-T Rec X.812 (1995) | ISO/IEC 10181-3:1996 "Security Frameworks for open systems: Access control framework".

[3] H. Debar, M. Dacier and A. Wespi, "Towards a taxonomy of intrustion-detection systems," Comput. Netw 31, Apr 1999, pp. 805-822.

[4] M. Serrano, S. Meer, J. Strassner, S. Paoli, A. Kerr and C. Storni, "Trust and Reputation Policy-Based mechanisms for Self-protection in Autonomic Communications," Proc. 6th International Conference on Autonomic and Trusted Computing, (ATC 09), Springer-Verlag, 2009, pp. 249-267.

[5] R. Sandu and J. Park, "Usage Control: A Vision for Next Generation Access Control," In Computer Network Security 2776, Springer-Verlag, 2003.

[6] ID:Analytics, White paper.: Analysis of Internal Data Theft (2008).

[7] A.P. Moore, D.M. Cappelli, T.C. Caron, E. Shaw, D. Spooner and R.F. Trzeciak, "A preliminary model of insider theft of intellectual property," In Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 2, 2011.

[8] BBC News. Siprnet: Where the leaked cables came from, http://www.bbc.co.uk/news/world-us-canada-11863618 [accessed 12-Jun-2011]

[9] D.W. Chadwick, G. Zhao, S. Otenko, R. Laborde, L. Su and T.A. Nguyen, "PERMIS: A modular Authorization Infrastructure", Concurrency and Computation: Practice and Experience 20, Aug. 2008, pp. 1341-1357.

[10] D.W. Chadwick, S. Otenko and T.A Nguyen, "Adding support to XACML for multi-domain user to user dynamic delegation of authority," International Journal of Information Security 8, Feb. 2009, pp. 137-152.

[11] Y. Brun, G.M. Serugendo, C. Gacek, H. Giese, H. Keine and M. Litoiu, "Engineering Self-Adaptive Systems through Feedback Loops," In Software Engineering for Self-Adaptive Systems, Springer-Verlag, 2009, pp. 48-70.

[12] J.O. Kephart and D.M. Chess, "The Vision of Autonomic Computing," Computer 36, Jan. 2003, pp. 41-50.

[13] J. Andersson, R. de Lemos, S. Malek and D. Weyns, "Modeling dimensions of Self-Adaptive software systems," In Software Engineering for Self-Adaptive Systems, Springer-Verlag, 2009, pp. 27-47.

[14] L. Shi and D.W. Chadwick, "A controlled natural language interface for authoring access control policies," Proc. ACM Symp. Applied Computing (SAC 11), ACM, 2011, pp. 1524-1530.

[15] A. Ferreira, D.W. Chadwick, P. Farinha, R. Correia, G. Zhao, R. Chilro and L. Antunes, "How to securely break into RBAC: the BTG-RBAC model," Proc. 2009 Annual Computer Security Applications Conference (ACSAC 09), IEEE Press, 2009, pp. 23-31.

[16] C. da Silva and R. de Lemos, "Dynamic plans for integrations testing of self-adaptive software systems," Proc. 6th International Symp. Software engineering for adaptive and self-managing systems (SEAMS 11), ACM, 2011, pp. 148-157.

[17] K. Böhm, S. Etalle, J. Den Hartog, C. Hütter, S. Trabelsi, D. Trivellato and N. Zannone, "A flexible architecture for privacy-aware trust management," Theoretical and Applied Electronic Commerce Research 5, Aug. 2010, pp. 77-96.