APPROXIMATING SAML USING SIMILARITY BASED IMPRECISION

Guillermo Navarro[1], and Simon N. Foley[2]

[1]*Dept. of Information and Communications Engineering,*
Universitat Autonoma de Barcelona, 08193 Bellaterra, Spain
gnavarro@ccd.uab.es

[2]*Dept. of Computer Science,*
University College, Cork, Ireland
s.foley@cs.ucc.ie

Abstract With the increasing complexity of networked systems has come the trade-off of security versus functionality; a strictly secured system is often an unusable system. As a consequence, users often entirely bypass security in order to get their job done. We consider how similarity techniques that are used by case-based reasoning systems can be used to provide a degree of control over how strictly/precisely security is enforced. The flexibility to be able to meaningfully control how strictly security is enforced is especially relevant in the emerging Web Services architectures, where a wide variety of different users and heterogeneous systems use a common framework to interoperate with a wide variety of different resources and services. The paper proposes *similarity-based imprecision security* (SBIS) for the *Security Assertion Markup Language* (SAML) as an approach to managing security in a web-services environment.

Keywords: Imprecise security, SAML, Case-Based Reasoning, access control.

1. Introduction

Traditional research on protection systems has focused on finding a system that can provide *absolute security*. That is, systems where only properly authorised actions can take place. By properly authorised we mean that the actions need to be absolutely classified, identified, authenticated, an so forth. Modern computer systems have become very complex. Many users may wish to interact and/or use many resources, which may be distributed across a network. Enforcing security across these systems becomes correspondingly complex and, if strictly enforced, can lead to an unusable system.

End users regularly fail to appreciate the security decisions that they must make because they barely understand the security policy in force, let alone how security mechanisms may interoperate. Users tend to deliberately ignore or bypass security to get their work done. For example, it is a common practice for users to deliberately share or disclose passwords to facilitate system access (Adams and Sasse, 1999). Large enterprises such as governments, academic centres or big corporations, have many formal rules and regulations. In practice, enterprises work by relying upon social networks and unwritten rules, which are often contrary to the written rules. After all, a strategy used by employees to pressure management in labour disputes is to *work to rule* (Odlyzko, 2003).

An example of the potential for complex security rules is the *Secure Assertion Markup Language* (SAML) that is used to express security information in Web Services and Grid. The use of overly strict security policies by Web Services will more than likely lead to security being bypassed by administrators in their effort to provide continuing service. We use approximation techniques from the area of Case Based Reasoning to provide a degree of control over how strictly a security policy is enforced. Rather than the conventional all-or-nothing security, our approach can be regarded as providing a security 'dial' that controls the degree of strictness of security enforcement that the system is willing to tolerate.

In this paper we describe how these approximation techniques can be used in a practical way in SAML based applications, such as web services. Section 2 motivates and describes related work. In Section 3 we describe similarity-based imprecise security. Section 4 describes how it is integrated with existing SAML-based frameworks and Sections 5 and 6 describe the extension of SAML to support imprecision information in practice. Section 7 concludes the paper.

2. Motivation and Related Work

Empirical studies reveal that security systems are failing to provide usable applications, from simple password-based systems (Adams and Sasse, 1999; Yan et al., 2004), to complex access control systems (Zurko and Simon, 1996). Existing research on the usability of security systems has mainly focused on user interfaces. While providing a better user interface may be an excellent solution for some systems, it is not necessarily the solution to more usable and secure systems (Whitten and Tygar, 1999; Smetters and Grinter, 2002).

In this paper we propose the use of a different approach to achieve more usability in security systems, by reconsidering how security decisions are taken. We consider an *imprecise* security system, where the decision engine can take into account the similarity between security related information. *Imprecise se-*

curity introduces more flexibility in protection systems. Imprecision is defined in terms of similarity of authorisation: for example, how similar is 'root' access in Unix to 'administrator' access in Windows? Rather than all-or-nothing security, imprecision provides degrees of security, which can be viewed as a a *security dial*. Turning the dial up, results in the system becoming closer to a very strict security system (in some sense more secure); turning the dial down relaxes security enforcement, permitting more imprecision (in some sense less secure). Some applications of *imprecise security* are:

- *Overcome complexity in large organisations*. Large organisations impose many administrative rules which can be counterproductive; employees deliberately bypass the rules to do their job. An absolute security system does not allow its employees to bypass the rules. To avoid bureaucracy, users stop using the system whenever possible, or else use it incorrectly (for example, sharing passwords or private keys).

- *Emergency situations*. Some emergency situations may require relaxing the security measures of a system. This security downgrading should be achieved in a fast and controlled way. For example, a doctor isolated in an hospital during a tropical storm, needs to access the records of a patient from another department, doctor or hospital.

- *Heterogeneous systems*. Interoperability of heterogeneous systems require the use of security information from one system to another, or reuse security policies between different environments. Due to the different nature and technologies of the different systems it may be impossible to provide an isomorphic mapping between them. Similarity measures provide degrees of imprecision that can allow a practical mapping to be defined.

Providing degrees of flexibility in security enforcement has been studied to a limited extent in the literature. In (Rissanen et al., 2004), the authors introduce the notion of *override* in access control policies. An access control request can be denied with the possibility of override. If the user agrees, the system allows the access under the audit of some authority. Our approach differs from this by providing, in effect, greater flexibility in defining how authorisations may be overridden. (Povey, 2000) proposes an *optimistic* security model that assumes that every access is legitimate and should be granted under the basis that the system can rollback illegitimate actions. We believe that in practice it would not be feasible to undo all illegitimate actions, and some minimum security should be provided. Nevertheless, while not following the dictum "Make the user ask for forgiveness not permission" (Blakley, 1996), our proposal does adopt some optimistic security model principles (see Section 3).

The research described in this paper builds on the suggestion in (Foley, 2002) that similarity measures could be used to provide imprecision in delegation for trust management systems. Supporting imprecision for information retrieval systems has been extensively considered in the literature on similarity-based retrieval for Case-Based Reasoning (CBR) systems (Aamodt and Plaza, 1994). Imprecision permits answers that may not formally meet the query condition, but can be considered 'close enough'. The contribution of this paper is a consideration of how imprecision techniques can be used in a practical setting, and in particular, how similarity can be usefully introduced to SAML and supported within Web Services and Grid architectures.

3. Similarity-based Imprecise Security Systems

We define a *similarity-based imprecision security* system or *SBIS* system, as a security system, where the security decisions take into account the similarity between permissions, attributes, authorisations, or other security-related information.

SBIS Characteristics

In general, a system that supports SBIS has the following characteristics.

- *Accountability*: given the imprecision supported by the system, there needs to be some guaranties of accountability. In SBIS, accountability may be achieved by strong authentication of the principals. Whether this authentication is provided by means of a centralised authority such as a PKI or not, will depend on the specific scenario, environment, and configuration of the system.

- *Auditability*: in order to provide accurate postmortem analysis of the system's operations. All operations permitted under similarity constraints should be logged in detail, so they can be analysed to study possible irregularities.

- *Constrained entry points*: as stated in (Povey, 2000), exceeding privilege should be a rarity, rather than a norm. If most of the actions need to be checked through the SBIS security check because they do not pass the absolute security check (see Figure 1), it is a symptom that the system is not properly set up.

- *Deterrents*: as in many access control systems, it is interesting to have mechanisms to punish principals who misbehave. This punishment can be economic or simply, restricting access permissions during a period of time (recall, "Make the user ask for forgiveness not permission").

- *Least intrusive*: one of the main ideas behind SBIS, is for it to be easy to integrate within existing security systems. SBIS systems can use any type of security related information, permissions, authorisations, security policies, rules, and so forth, as long as a similarity function is provided between them.

Similarity

Similarity is equivalent to the dual distance concept from a mathematical point of view, and has been successfully applied in CBR systems. There are several similarity function types and families, for instance, there are boolean, numeric or partial order (including *lattices*) functions. Boolean functions may be easily emulated with numeric functions if required, and a variety of numeric similarity values can be expressed or normalised to the domain $[0, 1]$, including $\mathbb{R} \cup \{-\infty, +\infty\}$. For the sake of simplicity, we consider only numeric similarity values. For a good review of lattice based metrics see (Osborne and Bridge, 1997).

Broadly speaking, a similarity function (denoted as '$\sim$') takes two arguments from a set of features P (permissions, attributes, etc.) and returns a similarity value:

$$\_ \sim \_ : (P \times P) \rightarrow [0, 1]$$

Similarity functions are reflexive ($x \sim x = 1 \; \forall x \in P$), and symmetric ($x \sim y = y \sim x \; \forall x, y \in P$).

The similarity function is applied to a concrete feature or to a set of features. In the CBR literature, there are some generic similarity functions such as the weighted nearest neighbour, induction, lattice based metric, or the similarity matrix. In some cases one can compose functions or create specific ones.

Table 1. Sample similarity matrix.

$\_ \sim \_$	*und*	*phd*	*prof*
und	1.0	0.7	0.2
phd	0.7	1.0	0.5
prof	0.2	0.5	1.0

For example, Table 1 defines a simple similarity matrix between three roles: undergraduate student (*und*), PhD student (*phd*), and professor (*prof*).

Similarity threshold

The *similarity threshold* is the threshold for which a given similarity value is accepted. We denote the similarity threshold as 'δ'. For instance, suppose an SBIS system where some action requires a permission p, but the user does

not hold such permission. The SBIS engine should look for a permission p' held by the user such that it is similar to p with a similarity value greater than the similarity threshold: $\exists p' | p \sim p' \geq \delta$.

The similarity threshold can be either *static* or *dynamic*. An *static similarity threshold* cannot be changed during execution time, while a *dynamic similarity threshold* can change its value during execution-time giving cause for a *security dial*.

4. Integration of SBIS in current SAML frameworks

An interesting issue is how an SBIS system could be integrated into an existing access control system, without having to introduce major modifications in the system. Our approach is to reuse the main components of an existing system. For instance, permissions, authorisations, attributes, and so on as provided in SAML assertions. We consider the use of SAML in a generic access control scenario as described in Figure 1. It involves a *Policy Decision Point* (PDP), which can take SAML assertions as the input to the access control decision, and a *Policy Enforcement Point*, which controls the access to a given service or resource.

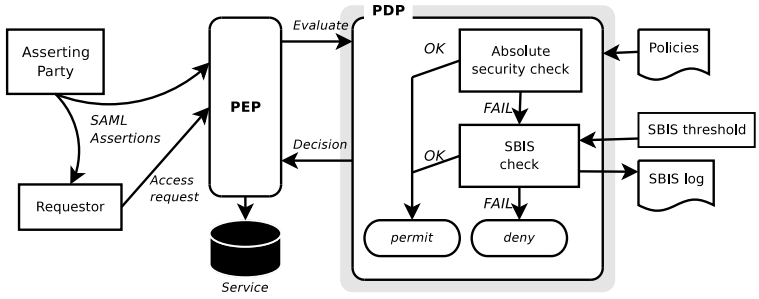

Figure 1. SBIS

Consider a set of SAML assertion statements P, with a partial order $\preceq$. Suppose that a given access request requires permission p and that the user making the request holds a set of statements Q. Then, the decision process could be summarised as:

Absolute security check: if $\exists q \in Q \mid q \preceq p$ then *permit*, otherwise SBIS check.
SBIS check: if $\exists q \in Q \mid q \sim p \geq \delta$ then *permit*, otherwise *deny*.

5. Expressing SBIS in SAML assertions

SAML provides a standard XML framework for exchanging security information between online business partners (OASIS, 2005). It is a well known standard applied in numerous industry products.

Security information is exchanged in form of *assertions*. SAML provides three types of *assertion statements*: Authentication, Attribute, and Authorisation Decision. Broadly speaking an assertion has an *issuer*, a *subject* or *subjects*, some *conditions* that express the validity specification of the assertion, and the *statement* (one or more). The assertion may be signed by the issuer.

Similarity-based assertion

We apply similarity functions to the SAML *statements* and their elements. For example, an authorisation decision includes the *resource* and the *action* of the authorisation. An authority may provide an authorisation decision assertion in terms of similarity between resources or actions, under some similarity threshold.

We introduce a new optional element contained by the SAML *statement* element called `SbisInfo` to provide SBIS-related information. Note that this information cannot be provided in the *condition* element of the assertion because it makes reference to the whole assertion, while a single assertion can provide more than one statement for the same subject.

```
<element name="sbis:SbisInfo" type="sbis:SbisInfoType"/>
<complexType name="sbis:SbisInfoType">
  <sequence>
    <element name="sbis:threshold" type="double"/>
    <element name="sbis:source" type="ID" minOccurs="0"/>
    <element name="sbis:function" type="sbis:SbisFunctionType" minOccurs="0"/>
    <element name="sbis:history" type="sbis:SbisHistoryType" minOccurs="0"/>
  </sequence>
</complexType>

<complexType name="sbis:SbisFunctionType">
  <sbis:element ref="cbml:similarity"/>
</complexType>
<complexType name="sbis:SbisHistoryTipe">
  <element ref="saml:AssertionURIRef" maxOccurs="unbounded"/>
</complexType>
```

Figure 2. SbisInfo XML Schema definition.

This *SbisInfo* element contains a sequence the following elements:

- *threshold*: the value of the threshold when the assertion was declared. If it is numeric, it will have to be normalised to the interval $[0, 1]$.

- *source*: optional element indicating the source of the similarity check, which is used to decide whether to trust the assertion or not. It will normally be the issuer of the assertion but this may not always be the case.

- *function*: the function used to calculate the similarity. This function is described using CBML (*Case Based Markup Language*) (Hayes and Cunningham, 1999), as described in Section 3.
- *history*: a sequence of URI references to assertions used to evaluate and issue this assertion. They may be used to verify the similarity constrains by a third party.

A simplified W3C's XML Schema definition of the *SbisInfo* element can be seen in Figure 2, which provides an extension of the current SAML assertion Schema (version 2.0). There, we denote the SAML namespace as *saml*, the CBML namespace as *cbml*, and the SBIS one as *sbis.*

```
<Assertion
    xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    Version="2.0"
    ID="http://www.library.edu/AuthenticationService/SAMLAssertions/126"
    IssueInstant="2005−04−13T16:30:00.173Z">
  <Conditions
      NotBefore="2005−04−13T16:30:00.173Z"
      NotOnOrAfter="2005−04−14T16:30:00.173Z"/>
  <Issuer>
    http://www.library.edu
  </Issuer>
  <Subject>
    <NameID NameQualifier="http://www.library.edu">
      Alice
    </NameID>
  </Subject>
  <AuthzDecisionStatement
      Resource="http://www.library.edu/PhDCatalog"
      Decision="Permit">
    <Action>read</Action>
    <SbisInfo>
      <threshold>0.5</threshold>
      <source>http://www.library.edu</source>
      <function>...</function>
      <history>...</history>
    </SbisInfo>
  </AuthzDecisionStatement>
  <ds:Signature>...</ds:Signature>
</Assertion>
```

Figure 3. SAML Assertion example.

Figure 3 shows an example of a simplified SAML authorisation assertion with SBIS information.

Expressing the similarity function

In order to express the similarity function, we use CBML, which is a generic XML-based language for CBR. This language can describe generic similarity functions (Coyle et al., 2004).

It is important to note the relevance of being able to express similarity functions in a common and standard way. An authority which provides some kind of attribute, can also provide the similarity function for its attributes. The authority will be the principal with more knowledge about the attribute, thus the expert who can provide the best similarity function. The similarity function, also serves as a proof to third parties of how the similarity was calculated. It includes the features used to calculate the similarity and how were they calculated.

6. Practical considerations

When a SAML asserting party or authority generates an assertion under similarity constraints the assertion includes the relevant information regarding the similarity threshold, similarity function and references to assertions used during the decision process. This information, together with the digital signature of the assertion is crucial for third parties that have to evaluate the assertion.

The SBIS information encoded within the assertion is sufficient to avoid the cascading problem (Foley, 2002). For example, given the example of the similarity function in Table 1, for the roles: undergraduate student (*und*), PhD student (*phd*), and professor (*prof*). Consider that Alice has an attribute assertion issued by a recognised authority *Chancellor*. If we consider a decision point with a similarity threshold $\delta = 0.5$, Alice will be able hold the *phd* role ($und \sim phd = 0.7$), but not the *prof* role ($und \sim prof = 0.2$). In some situations Alice may ask the decision point to issue a new assertion stating that she can hold the role *phd* to use the assertion in another access request for example. But note that then, she could use such assertion to gain privileges for the role *prof*, since $phd \sim prof = 0.5$. While this is a simplistic example, the cascading problem must be dealt with carefully in more complex situations. To avoid cascades, when Alice asks the decision point to issue the second assertion, this assertion will have all the information previously commented. Thus, the receiver is able to evaluate and track the similarity constraints used to issue the assertion.

In (Foley, 2002) a specific solution for the cascading problem is provided. While providing an elegant solution, it requires an *a priori* knowledge of the similarity functions involved in the similarity-based decision (including future ones). In this paper we provide a more generic solution. One could use the assertion to compare it to a new attribute assertion (such as age) and provide a new different similarity function.

7. Conclusions

This paper discusses the relevance and motivation for allowing a controlled degree of imprecision in security decision engines. This degree is based on the

similarity between security related information such as permissions, attributes, etc. resulting in *similarity based imprecision security* (SBIS) systems. We introduced SBIS capabilities in SAML, which is a widely adopted standard in Web Services and Grid.

SBIS is not intended for high security or critical systems but systems where usability is a key point. Empirical studies demonstrate that currently, absolute security systems are failing to do their job. SBIS can provide enough flexibility to the system and at the same time it can ensure some degree of security.

Acknowledgments

The work of G. Navarro has been partially funded by the Spanish Ministry of Science and Technology (MCYT) though the project TIC2003-02041.

References

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AICom- Artificial Intelligence Communications*, 7(1).

Adams, A. and Sasse, M. A. (1999). Users are not the enemy. *Commun. ACM*, 42(12).

Blakley, B. (1996). The emperor's old armor. In *Proceedings of the 1996 workshop on New security paradigms*.

Coyle, L., Doyle, D., and Cunningham, P. (2004). Representing similarity for CBR in XML. In *Advances in Case-Based Reasoning (Procs. of the Seventh European Conference)*.

Foley, S. N. (2002). Supporting imprecise delegation in keynote using similarity measures. In *Proceedings of International Security Protocols Workshop*.

Hayes, C. and Cunningham, P. (1999). Shaping a CBR view with XML. In *Proceedings of the Third International Conference on Case-Based Reasoning and Development, ICCBR-99*.

OASIS (2005). Assertions and Protocols for the OASIS Secure Assertion Markup Language (SAML) v2.0. sstc-saml-core-2.0-cd-04, Committee Draft 04.

Odlyzko, A. (2003). Economics, psychology, and sociology of security. In *Financial Cryptography: 7th International Conference*.

Osborne, H. and Bridge, D. (1997). Models of similarity for case-based reasoning. In *Procs. of the Interdisciplinary Workshop on Similarity and Categorisation*.

Povey, D. (2000). Optimistic security: a new access control paradigm. In *Proceedings of the 1999 Workshop on New Security Paradigms*.

Rissanen, E., Firozabadi, B. Sadighi, and Sergot, M. (2004). Towards a mechanism for discretionary overriding of access control. In *12th International Workshop on Security Protocols*.

Smetters, D. K. and Grinter, R. E. (2002). Moving from the design of usable security technologies to the design of useful secure applications. In *Proceedings of the 2002 Workshop on New Security Paradigms*.

Whitten, A. and Tygar, J. D. (1999). Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium*.

Yan, J., Blackwell, A., Anderson, R., and Grant, A. (2004). Password memorability and security: Empirical results. *IEEE Security & privacy*, 2(5).

Zurko, M. E. and Simon, R. T. (1996). User-centered security. In *Proceedings of the 1996 Workshop on New Security Paradigms*.