Triggering IDM Authentication Methods based on Device Capabilities Information

Marcus Quintino Kuhnen¹, Mario Lischka², and Félix Gómez Már mol³

¹marcus.kuhnen@googlemail.com
²AGT Group (R&D) GmbH, Darmstadt Germany, mlischka@agtgermany.com
³NEC Laboratories Europe, Heidelberg Germany, felix.gomez-marmol@neclab.eu

Abstract: Identity management systems are a reality today in the Internet. Single sign-on (SSO) systems allow users to authenticate once in the system and interact with different services providers without the need for creating new accounts. However, most identity management systems only support a simple authentication mechanism, which most of the cases is based on login and password, with its well known associated vulnerabilities like phishing attacks, for instance. In order to mitigate those drawbacks and improve the overall security of the system, we propose an enhancement of SSO systems which allows the identity providers to dynamically choose the best authentication method (e.g. fingerprint, digital certificates, smart cards, etc) being applied to the user based on the users’ device capabilities and context information.

1 Introduction

Nowadays, fixed and mobile networks are converging and thus a new wave of services are being offered to the users. As a side effect, the amount of information that a user has to manage is being steadily growing. In particular for each new service a new account is created and crucial personal and financial information have to be shared many times. Along the past years, various Identity Management (IdM) systems have been developed and deployed. Their technology should enable the user to easily interact with diverse services that are being offered today, especially at the e-Commerce scope.

Users of an IdM system do not need to have a login and password for each Web Service they want to access when single sign-on is supported. Using this identity model, the service provider just needs to have a trust relationship with the IdM system [GMKMP11]. In this way, the authentication and authorization for the usage of the service can be always mediated by the IdM system.

However, although IdM systems make things look easier from the users’ and service providers’ perspective, there is a major drawback. A user-centric SSO system usually assumes that all data related to the user’s credentials has to be stored centralized in one single repository, therefore opening the doors to several types of attacks. Hence, if the

*This paper is based on the authors work while being employed at NEC Laboratories Europe
user is victim of a phishing attack [JJIM07], for instance, the attacker will not steal only one password, but rather an agglomerate of private user’s data, therefore giving him access to all the user’s service accounts. In [WDRJ10] authors show that users prefer those authentication methods which are better known, even when they might not be as secure as other choices. [Bri08] shows that new methods of authentication like smart cards, fingerprint biometrics and converged cards have been evaluated. It also underscores the strategic value of solutions that can effectively deploy and manage multiple authentication methods. Thus, strong authentication and its different methods is an unquestionable feature that IdM systems should support.

Consequently, in this paper we present an enhancement of IdM systems that enables Identity Providers (IdP) with the ability of dynamically choosing the best authentication mechanism by detecting the capabilities of the device the user is utilizing for authenticating herself, as shown in Figure 1. These devices ranges from desktop computer and laptops over tabloid PCs to smart phones.

![Figure 1: Scenario description](image)

The remainder of this paper is organized as follows: Section 2 describes some of the current related works in this field. In Section 3 we present a use case where to apply our solution. An overview of such solution is introduced in Section 4 and a more detailed description in Section 5. The application to the aforementioned use case is presented in section 6. We conclude with some final remarks in Section 7.

## 2 State of the Art

In current Identity Management systems such as those based on SAML, OpenID or frameworks like SASL [MZ06], the service provider specifies the authentication methods which the identity provider should use to authenticate the user according to its predefined policies and level of assurance [BDP06]. Usually, this authentication method is indicated as a specific authentication class. For example, the SAML core specification [OAS05a] indicates that such authentication classes enable the authentication authority and relying
party to come to an agreement on what are the acceptable authentication classes by giving them a framework for negotiation [OAS05b]. In a similar manner, an extension to the OpenID protocol called OpenID Provider Authentication Policy Extension has been proposed [Ope07]. Nevertheless, these frameworks do not consider a further communication with the user equipment that could have a decisive role in the process. Moreover, they do not provide a mechanism to adapt from weak to strong authentication depending on context information available, such as devices capabilities of the user and internal knowledge about the user (e.g. additional communication channels, list of previously provided tokens, etc.). The same is true for the service providers: they do not interact with the user in order to choose the most appropriated method for authentication. Therefore, the choice of an authentication method of current state of the art systems does not consider the context and devices of the user.

Some authentication mechanisms for mobile phones have been examined in [Vap10] regarding their resistance against certain threats. In [DSBL10] authentication and payment methods via web browsers are presented which utilizes QR codes that are read and analysed by a smart phone, which does the final steps. In [EFJr09] presents the criteria for the evaluation of authentication schemes in IMS, which include security, user-friendliness and simplicity. However, the specific authentication methods provided through the phone hardware have not been taken into account.

3 Use Case

In order to highlight the importance of a dynamic authentication method considering the device capabilities of the user in order to trigger dynamically different authentication methods, let’s consider the following scenario: A user wishing to access a service provider could have the different types of hardware capabilities to provide authentication credentials, e.g., finger print, ability to send SMS messages.

Based on the current state of the art [OAS05b, Ope07], a service provider prompting the user for an authentication procedure would request for a fixed procedure based solely on its authentication policies. However, the end user might be equipped with a finger print device on her laptop or her mobile phone. Such devices actually allow a much more secure authentication than login and password mechanisms, since the user is able to access the service using her mobile phone and use additional mechanisms, e.g. SMS, for sending their credentials. Alternatively, the user’s finger print could be used for authentication.

Therefore, if the service provider would have this information, it could enable a much more flexible mechanism to match service providers authentication policies regarding the level of service assurance, and what can be offered by the capabilities of the user’s devices.
4 Solution overview and architecture

Regarding the authentication of their users in a federated environment, the service providers (SP) interests are two-fold: one the one hand they want authentication methods in place which match their security needs; one the other hand the user should be able to use their service from a wide range of devices.

Figure 2 shows the extended authentication protocol which is utilizing the well-known single-sign on pattern extended by the additional step 4, 5 and 6. Once the user equipment (UE) interacts with the IdP, the UE sends its device capabilities based on the IDP device capabilities request.(step 4 and 5). Based on the requirements provided by the SP in steps 2 and 3 the IdP chooses the best matching authentication method (step 6) and sends the corresponding reply to the UE (step 7). Assuming a successful authentication (step 8), the UE is then redirected to the SP with a corresponding assertion on the identity including the authentication method used.

Figure 2: Discovering Device Capabilities

In order to provide the necessary capabilities we propose that an additional plugin is deployed at the UE. This signed browser plugin accesses the hardware of the device and issues the response with the device capabilities information. This plugin should, in addition, enable the triggering of the right authentication method among the ones provided by the particular device.

The decision on which authentication method is the most appropriate is controlled through XACML policies [OAS09]. We choose this policy language since it allows us to directly retrieve information from XML files which encode the capabilities. Moreover, it has a high expressiveness regarding the conditions which have to be fulfilled to choose a method. Due to the extensions in version 3, obligations provide a flexible method to trigger additional functions such as authentication methods or redirects.
5  Solution details

In this section we will present the details of our approach regarding the encoding of the device capabilities, the determination of the authentication algorithm through XACML mechanisms, and an analysis of potential threats.

5.1  Description of the device capabilities

As we already presented in [KKS+09], a device can describe its capabilities using the Composite Capabilities/Preference Profiles (CC/PP) based User Agent Profile (UAprof [Ope06]) standard. This standard defines six categories of device capabilities: hardware, software platform, browser user agent, Network, WAP and Push characteristics. A snapshot of this document can be observed in listing 1.

Since this is a mature specification, we can rely on it for analyzing the devices’ capabilities. Moreover, we propose to enhance the RDF based device capabilities description using the new bag <rdf:AuthenticationSupport> element, which can be further extended to any number of authentication types. While a standardized list as the one in [OAS05b] might be used, instead of the free text in the example listing 1, additional new methods have to be known between the UE plugin and the IdP.

This information, together with the mechanism to be described in section 5.2, can provide IdM systems with a more enhanced authentication method by allowing to personalize the authentication process, based on users’ device and authentication capabilities.

According to the proposed extension points in SAML standard protocol we embed the RDF document in a SAML response, in reply to a SAML request issued by the IdP to the user about her device capabilities.
5.2 XACML mechanism and Controlling the authentication method

The second aspect is related to the XACML policy engine [OAS09] of the Identity Provider. Based on the mechanisms of the XACML language, the selection of the authentication method is based on the device’s capability descriptions shown in previous section. This way the best authentication mechanism available for the needs of the service provider on the type of device that the user is currently using is selected and applied.

The policy decision point (PDP) receives a detailed request which contains all the information necessary to make a decision. While we argued in [HKK+09] that the whole capability list would be too large to be included in each request (which in that approach was quite frequent on a number of SIP events) in this solution the situation is different, since the decision has to be made only when the user has to be authenticated at the first time, or a re-authentication is required due to an expiration, major-context switch, or a higher level of authentication security is required.

Thus, instead of requesting an attribute utilized in a policy each time via the policy information point (PIP), we include the capability list in the request. According to the standard, the element ResourceContent allows the inclusion of an XML document in the request sent to the decision engine. The element AttributeSelector enables the editor of a policy to refer to specific values in the XML document via XPATH expressions.

Besides the capability descriptions of the user’s device, the request contains the attributes shown in table 1, where xs-adc is just a prefix to ensure unique names. While the serviceProvider will be assigned to the access–subject category of XACML3, the other attributes are in the resource category.

<table>
<thead>
<tr>
<th>AttributeID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xs-adc:serviceProvider</td>
<td>the requesting service provider, who triggers the selection</td>
</tr>
<tr>
<td>xs-adc:authenticationLevel</td>
<td>the level of security targeted for authentication</td>
</tr>
<tr>
<td>xs-adc:assumedUserID</td>
<td>the identity to be authenticated, if available beforehand</td>
</tr>
</tbody>
</table>

Table 1: Attributes used in XACML request

In addition to the actual effect of an XACML policy (“Permit” or “Deny”), more detailed actions could be defined by adding obligations (one of the extensions of the latest XACML version 3) to policy sets, policies and rules. The main idea is that the obligations should trigger the most appropriated authentication method according to the user’s device capabilities and level of assurance requested by the Service Provider. However, if such level of assurance cannot be met by matching the device capabilities and the level of assurance, the rule mechanism could express that the user should be automatically redirected to another Identity Provider in the federation that can apply the requested level of assurance.

We assume that the request for authentication will be permitted in case a potential authentication method is available. In case no authentication method is available at the IdP which is controlled through these policies, then the request is denied (although a redirection to a different IdP is additionally possible).

In addition to the permit or deny decision, the authentication method which should be
used has to be provided to the policy enforcement point (PEP). This is done by including an obligation in the XACML response. We assume two types of obligations: 

\texttt{authenticate(method)} and \texttt{redirect(URI, method)}.

The first obligation contains a unique identifier for the method to be used. The second obligation included in the XACML response is used in case the user gets a “Deny”, in order to forward the authentication request to another Identity Provider which is known to be able to authenticate the user with the specific method. In case the second parameter is left empty, then the other Identity Provider has to decide on its own which method should be used.

5.3 Threats Analysis

The proposed extensions enables the user entity to influence the authentication method chosen by the identity provider. At a first glance this might be a new security threat in case the user entity got corrupted or the communication is unsecure. In the first case one has to observe that the list of capabilities is always matched with the one provided by the SP, and the SP specifies what is the minimal authentication method to be used\(^1\). Thus a corrupted UE might indicate that it is only capable to provide a very weak authentication method, but if this is not acceptable, the whole authentication process will fail. Regarding communication security and a potential man-in-the-middle attack in order to manipulate the capabilities of a device, the plugin could generate a public key pair which will be used to sign its list of capabilities\(^2\).

Deploying a plugin software for the UE in a secure and trustworthy way is out of the scope of this paper. One has to observe that the software would be (in-) directly be provided by the IdP, to which the user has a stronger (trust) relationship, compared to other service providers. Although this also implies a trustworthy PKI to be in place which has additional challenges and threats, we see this as one of the preconditions not only for a secure software distribution in the future.

6 Application to a Use Case

As described in section 3, we could imagine a service provider requiring an authentication method which is highly secure. Since the user is currently requesting the service through her laptop a finger print reader as well as a camera providing a life picture of the user are available. Thus, the related request contains the service provider, the authentication level and the device capabilities. The excerpt of a policy shown in listing 2 ensures that, depending on the available capabilities, the finger print reader is chosen, if present. If not,

\(^1\)Additional signatures may ensure that no alteration is possible.

\(^2\)As this could be done offline, even a weak device could precompute the signature and send the signed list without any delays
the OneTimePassword Token ("OTP") authentication method is chosen\(^3\).

The rule combining algorithm first applicable (see line 1) ensures that the first match will determine the decision and the obligation of the rule which have to be fulfilled. The target section of the policy (line 2) ensures this policy is only evaluated based on a request issued by service provider YourBank for the authentication with level3. Other service providers will be handled by different policies. Narrowing the target of a policy ensures a better performance by avoiding more costly function calls used in the conditions.

Listing 2: XACML Policy

```xml
<Policy xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  RuleCombiningAlgId="...:rule-combining-algorithm:first-applicable" ...
  >
  <Target> <AnyOf> <AllOf> 
  <Match MatchId="urn:oasis:names:tc:xacml:1:0:function:string-equal">
    <AttributeValueDataType="...#string">YourBank</AttributeValueDataType>
    <AttributeDesignator MustBePresent="false" Category="urn:oasis:names:tc:xacml:1:0:subject-category:access-subject" AttributeId="xs-adc:serviceProvider" DataType="...#string"/>
  </Match>
  <Match MatchId="urn:oasis:names:tc:xacml:1:0:function:string-equal">
    <AttributeValueDataType="...#string">level3</AttributeValueDataType>
    <AttributeDesignator MustBePresent="false" Category="urn:oasis:names:tc:xacml:3:0:attribute-category:resource" AttributeId="xs-adc:authenticationLevel" DataType="...#string"/>
  </Match>
  </AnyOf> <AllOf>
  <Rule Effect="Permit"> ... <Target> <Condition>
  <Apply FunctionId="urn:oasis:names:tc:xacml:1:0:function:string-is-in">
    <AttributeSelector Category="urn:oasis:names:tc:xacml:3:0:attribute-category:resource" Path="ccpp:component/AuthenticationSupport/text()" DataType="...#string"/>
    <AttributeValueDataType="...#string">FingerPrint</AttributeValueDataType>
  </Apply> ...
  </Condition>
  <ObligationExpressions>
  <ObligationExpression ObligationId="xs-adc:authentication" FulfillOn="Permit">
    <AttributeAssignmentExpression AttributeId="xs-adc:methode">
      <AttributeValue DataType="...#string">FingerReader</AttributeValueDataType>
    </AttributeAssignmentExpression>
  </ObligationExpression>
  </ObligationExpressions>
  </Rule> <Target> <Condition>
  <ObligationExpressions>
  <ObligationExpression ObligationId="xs-adc:redirect" FulfillOn="Deny">
    <AttributeAssignmentExpression AttributeId="xs-adc:URI">
      <AttributeValue DataType="...#string">SecondaryIdP</AttributeValueDataType>
    </AttributeAssignmentExpression>
  </ObligationExpression>
  </ObligationExpressions>
  </Rule> <Policy>
```

The retrieval of the supported authentication method from the request could not be done in the target section of XACML. Thus, the bag rdf:AuthenticationSupport is selected with an XPATH expression (line 11) in the condition of the rule.

\(^3\)The handling of the OneTimePassword Token ("OTP") is not shown for simplicity.
The function \textit{string-is-in} (line 10) checks whether the string \textit{FingerPrint} is contained in the selected bag. If this condition (and potential additional ones like time constraints, etc) holds, then the related obligation (line 14) will be evaluated and returned. The method indicated in the obligation will then be executed as step 8 in Figure 2.

The final rule (line 19) of the example policy in listing 2 ensures that the request is denied and forwarded to another identity provider due to the obligation in line 21.

This example gives a first impression on how XACML could be used to determine the authentication method to be used. More complex conditions like time constrains (the desktop could not be used outside of the office hours), or checking for failed accesses, for instance, could be easily added to the specific condition elements.

7 Conclusions

IdM systems are widely spread nowadays in the Internet due to their numerous advantages for both end users and service providers. One of those nice facilities is the SSO feature, allowing users to be authenticated once while having access to different services.

Yet, many web services today use the username and password as authentication method, with its known risks, like the phising attack, for instance. As a consequence an attacker obtaining the username and password of a victim will get access to several services.

But more secure authentication mechanisms like fingerprint, smart cards or digital certificates are a reality too. Hence, in the paper at hand we have proposed to enhance the IdM systems in order to dynamically choose the most appropriate authentication mechanism, based on the capabilities of the device the user is employing to authenticate herself, from a well-defined list of available authentication methods.

Our approach entails that the IdPs have to find out which are the authentication capabilities of the device through whom the user is requesting the service. To do so, a request is sent to the user equipment, who replies with a response, containing such capabilities described using the RDF language, e.g as an extension of the SAML protocol. Then, the decision on which authentication mechanism should be finally used is taken by means of XACML policies, avoiding the threat that an undermined user client is triggering only weak authentication methods. Depending on the application scenarios registration of the devices and their capabilities will allow the IdP to detect spoofed messages.

Our solution applies well known current technologies and it is therefore easy to implement and deploy. We are convinced that this solution will contribute to increase the overall security in the authentication processes taking place within the IdM systems in the Internet today.
References


[Ope06] Open Mobile Alliance. User Agent Profile, February 2006. OMA-TS-UAProf-V2_020060206-A.

