ABFAB
Internet-Draft
Intended status: Informational
Expires: April 7, 2016

A. Perez-Mendez
R. Marin-Lopez
G. Lopez-Millan
University of Murcia

October 5, 2015

Retrieving remote attributes using GSS-API naming extensions
draft-perez-abfab-gss-remote-attr-00

Abstract

The GSS-API Naming Extensions define new APIs that extend the GSS-API naming model to support name attribute transfer between GSS-API peers. Historically, this set of functions has been used to obtain the authorization information contained in some sort of authorization token provided to the GSS acceptor during the context establishment process, such as a Kerberos ticket, a SAML assertion, or an X.509 attribute certificate. However, some scenarios require to allow the GSS acceptor to request additional attributes after context establishment. If these attributes are not locally stored by the GSS mechanism they have to be retrieved from an external source (e.g. SQL database, LDAP directory, external IdP, etc.). This document describes how current GSS-API extensions are able to encompass such functionality without requiring of any change, neither on the existing calls nor on the way applications use the API.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 7, 2016.

Table of Contents

1. Introduction

The Generic Security Service Application Programming Interface (GSS-API) Naming Extensions [RFC6680] define new APIs that extend the GSS-API naming model to support name attribute transfer between GSS-API peers. Historically, this set of functions has been used to obtain the authorization information contained in some sort of authorization token provided to the GSS acceptor during the context establishment process, such as a Kerberos ticket, a SAML assertion, or an X.509 attribute certificate. Therefore, after context establishment, the GSS mechanism could provide any of the available name attributes to the application (GSS acceptor) without requiring any further interaction with any external entity.

However, some scenarios require to allow the GSS acceptor to request additional attributes after context establishment. In those cases, it is possible that these attributes are not locally stored by the GSS mechanism and, therefore, they have to be retrieved from an

external source (e.g. SQL database, LDAP directory, external IdP, etc.). This document describes how current GSS-API extensions are able to encompass such functionality without requiring of any change, neither on the existing calls nor on the way applications use the API.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Motivation

This document is motivated by the work being done in the IETF ABFAB WG. In particular, the ABFAB architecture [I-D.ietf-abfab-arch] defines a way by which a federated end user can authenticate with a particular application by using a new GSS-API mechanism called GSS-EAP [RFC7055]. This mechanism conveys EAP packets between the end user (acting as EAP peer and GSS initiator), the application (acting as EAP authenticator and GSS acceptor), and the RADIUS server (acting as EAP server). Moreover, the ABFAB architecture also defines how a SAML assertion containing information about the end user is delivered to the GSS acceptor after the successful authentication of the end user. This latter process is described in the ABFAB Authentication Profile of [I-D.ietf-abfab-aaa-saml].

To this point, the GSS-EAP mechanism can make the information contained in the SAML assertion available to the application, by using the standard GSS-API Naming Extensions calls [RFC7056]. That is, the GSS_Inquire_Name() call returns the list of attributes available in the SAML assertion, whereas the GSS_Get_Name_Attribute() one returns the value of each specific attribute.

However, [I-D.ietf-abfab-aaa-saml] also defines the ABFAB Assertion Query/Request Profile. Using it, the GSS acceptor can send

additional SAML Attribute Queries to the IDP after the end user has been authenticated, in order to obtain attributes that were not obtained before (i.e. not listed by GSS_Inquire_Name()). It was not clear for the ABFAB WG whether this behaviour was possible using current GSS-API calls, so this document intends to shed some light on the subject.

4. General operation

Despite its typical usage, the GSS-API naming extensions document [RFC6680] takes no assumption on how a particular GSS mechanism obtains the attributes associated to a name. Indeed, as any other

API, the GSS-API works as a "black box", where only inputs and outputs are specified. Hence, it is irrelevant whether the GSS mechanism has the attributes locally stored, or it has to retrieve this information from an external source. Moreover, [RFC6680] does not forbid the usage of the GSS_Get_Name_Attribute() call to request the value of an attribute that is not listed in the output of the GSS_Inquire_Name() call.

Taking the previous considerations into account, the following steps describe a proposal by which any GSS-API mechanism implementation can retrieve remote attributes and provide them to the application in a transparent way:

1. The application gets the list of attribute names by invoking GSS_Inquire_Name().

2. The application is interested on a particular attribute (called "ATTR") that is not included in the results from step 1.

3. The application requests the value of attribute "ATTR" by invoking GSS_Get_Name_Attribute().

4. The GSS mechanism processes the request and realizes the attribute is not available into its local storage. Instead of generating an GSS_S_UNAVAILABLE error, the mechanism checks its configuration to see whether it can retrieve the attribute from an external source (e.g. SQL database, LDAP directory, SAML IDP...). There are two possible outcomes for this process:

a. If the attribute is successfully retrieved from the remote source, the GSS mechanism adds it to the local storage so it will be listed by future invocations of GSS_Inquire_Name(). This process is similar as if the GSS_Set_Name_Attribute() would have been called with the retrieved attribute data. Finally, the attribute value is provided to the application as if the attribute would have been located into the local storage. Note that the application is unaware of this whole process.

b. If the attribute cannot be retrieved, the GSS mechanism returns the GSS_S_UNAVAILABLE error major code. The minor code might provide additional information (e.g. connection refused, timeout, invalid SAML attribute format...) if desired.

5. The application makes use of the obtained attribute, or take the associated countermeasures if the attribute was not provided.

5. Example using draft-ietf-abfab-aaa-saml, GSS-EAP and RFC7056

The following example follows the steps described in the previous section, with actual examples using draft-ietf-abfab-aaa-saml, GSS-EAP and [RFC7056].

1. The application gets the list of attribute names by invoking GSS_Inquire_Name(). The list returns the following attribute names:

   * urn:ietf:params:gss:radius-attribute 79 (RADIUS EAP-Message)

   * urn:ietf:params:gss:radius-attribute 80 (RADIUS Message-Authenticator attribute)

   * urn:ietf:params:gss:radius-attribute 1 (RADIUS User-Name attribute)

   * urn:ietf:params:gss:radius-attribute 24 (RADIUS State attribute)

   * urn:ietf:params:gss:federated-saml-assertion (Complete SAML

assertion)

* urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:1.3.6.1.4.1.5923.1.1.1.7 (SAML eduPersonAffiliation attribute)

These attribute names follow the format described in [RFC7056].

2. The application is interested on getting the value of a standard federated SAML attribute called eduPersonAffiliation. The format of the attribute name following [RFC7056] is "urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:1.3.6.1.4.1.5923.1.1.1.1".

3. The application requests the value of the attribute by invoking GSS_Get_Name_Attribute().

4. The GSS-EAP mechanism implementation processes the request, and realizes the attribute is not available into its local storage. Then, it generates a new SAML Attribute Query for that attribute, sending it to the IdP using the ABFAB Assertion Query/Request Profile described in section 8 of [I-D.ietf-abfab-aaa-saml]. As a result, it obtains a SAML Response with a SAML Attribute Statement for the requested attribute. Finally, the GSS-EAP

mechanism implementation stores the attribute into its local storage, and provides the output of the GSS_Get_Name_Attribute() call with the value of the attribute.

5. The application uses the attribute to perform authorization tasks.

6. If the application invokes GSS_Inquire_Name() again, the result would contain the following attribute names:

* urn:ietf:params:gss:radius-attribute 79 (RADIUS EAP-Message)

* urn:ietf:params:gss:radius-attribute 80 (RADIUS Message-Authenticator attribute)

* urn:ietf:params:gss:radius-attribute 1 (RADIUS User-Name attribute)

* urn:ietf:params:gss:radius-attribute 24 (RADIUS State attribute)

* urn:ietf:params:gss:federated-saml-assertion (Complete SAML assertion)

* urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:1.3.6.1.4.1.5923.1.1.1.7 (SAML eduPersonAffiliation attribute)

* urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:1.3.6.1.4.1.5923.1.1.1.1 (SAML eduPersonEntitlement attribute)

6. Considerations on blocking calls

The fact of performing network interactions within the implementation of an API call may lead to blocking situations, where the GSS mechanism needs to wait for a third entity to provide a result. This behaviour is allowed by current specification [RFC6680], although it may not be foreseen by some applications that (wrongly) assume GSS-API calls will return immediately.

After a discussion on the Kitten WG mailing list [KITTENDIS], it was concluded that it would be reasonable for an application to assume that the calls to GSS_Inquire_Name() and to GSS_Get_Name_Attribute() for attributes listed by the former will be faster (i.e. non-

blocking) than the calls to GSS_Get_Name_Attribute() for non-listed attributes (i.e. require network interaction).

Furthermore, it would be desirable that new definitions of GSS naming attributes explicitly declare whether a network interaction is possible to be triggered, so applications are aware that the results may take some time to be returned.

Finally, this document recommends GSS mechanism to implement timers to prevent a call to GSS_Get_Name_Attribute() keeps blocked if the external entity does not reply in a reasonable amount of time. If the timer expires, GSS_Get_Name_Attribute() returns with the GSS_S_UNAVAILABLE code, as if the attribute was not available at all. The returned minor code might provide some information about the actual cause of the error (i.e. timeout).

7. IANA Considerations

There are no IANA Considerations.

8. Acknowledgements

Authors thank JISC for their support.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6680] Williams, N., Johansson, L., Hartman, S., and S. Josefsson, "Generic Security Service Application Programming Interface (GSS-API) Naming Extensions", RFC 6680, DOI 10.17487/RFC6680, August 2012, <http://www.rfc-editor.org/info/rfc6680>.

9.2. Informative References

[RFC7055] Hartman, S., Ed. and J. Howlett, "A GSS-API Mechanism for the Extensible Authentication Protocol", RFC 7055, DOI 10.17487/RFC7055, December 2013, <http://www.rfc-editor.org/info/rfc7055>.

[RFC7056] Hartman, S. and J. Howlett, "Name Attributes for the GSS-

API Extensible Authentication Protocol (EAP) Mechanism", RFC 7056, DOI 10.17487/RFC7056, December 2013, <http://www.rfc-editor.org/info/rfc7056>.

[KITTENDIS]
Perez-Mendez, A., "Use of GSS_Get_name_attribute() to obtain further attributes. https://www.ietf.org/mail-archive/web/kitten/current/msg05550.html", April 2015.

[I-D.ietf-abfab-arch]
Howlett, J., Hartman, S., Tschofenig, H., Lear, E., and J. Schaad, "Application Bridging for Federated Access Beyond Web (ABFAB) Architecture", draft-ietf-abfab-arch-13 (work in progress), July 2014.

[I-D.ietf-abfab-aaa-saml]
Howlett, J., Hartman, S., and A. Perez-Mendez, "A RADIUS Attribute, Binding, Profiles, Name Identifier Format, and Confirmation Methods for SAML", draft-ietf-abfab-aaa-saml-11 (work in progress), August 2015.

Authors' Addresses

Alejandro Perez-Mendez
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia 30100
Spain

Phone: +34 868 88 46 44
EMail: alex@um.es

Rafa Marin-Lopez
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia 30100
Spain

Phone: +34 868 88 85 01
EMail: rafa@um.es

Gabriel Lopez-Millan
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia 30100
Spain

Phone: +34 868 88 85 04
EMail: gabilm@um.es