ABFAB
Internet-Draft
Intended status: Experimental
Expires: December 3, 2011

A. Perez-Mendez
R. Marin-Lopez
F. Pereniguez-Garcia
G. Lopez-Millan
University of Murcia
Jun 2011

GSS-EAP pre-authentication for Kerberos
draft-perez-abfab-eap-gss-preauth-00

Abstract

This draft defines an alternative to the standard cross-realm operation in Kerberos, to allow users from an organization can obtain a TGT from the KDC of a different one, both belonging to the same AAA-based federation. This proposal makes use of the GSS-API pre-authentication for Kerberos and the GSS-API Mechanism for the Extensible Authentication Protocol to carry out the required functionality.

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 December 3, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

Kerberos [RFC4120] is becoming one of the most widely deployed protocols for authentication and key distribution, as it is integrated in different operating systems and network applications (FTP, SSH, HTTP...). However, Kerberos usage is typically used to control the access of the subscribers of a single organization, since Kerberos multi-domain (cross-realm) infrastructures are not usually deployed. This draft aims to provide an alternative to the typical cross-realm operation in Kerberos by making use of existing Authentication, Authorization and Accounting (AAA) infrastructures, the Extensible Authentication Protocol (EAP) [RFC3748] for authentication and the SAML [OASIS.saml-core-2.0-os] and XACML [OASIS.xacml-2.0-core-spec-os] for authorization.

Since organizations typically deploy AAA infrastructures for controlling the access to services (especially network access service) in federated networks, the lack of a correct integration between Kerberos and AAA infrastructures limits the service access based on Kerberos only to service provider's subscribers, defeating the purpose of the network federations: to allow any end user in the federation to access any service deployed within it. In this document, we define a unified architecture to integrate existing AAA infrastructures with the service access control based on Kerberos. Specifically, we propose the user to perform a pre-authentication process with the visited KDC based on EAP, combining the use of the the GSS-API pre-authentication for Kerberos [I-D.perez-krb-wg-gss-preauth] and the GSS-API Mechanism for the Extensible Authentication Protocol [I-D.ietf-abfab-gss-eap]. Additionally, this solution also introduces authorization management based on the SAML and XACML standards.

1.1. Requirements Language

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].

2. Elements of the architecture

Brief description of the elements in the proposed architecture.

- End User. Integrates the functionality of Kerberos client, GSS initiator and EAP peer.

- KDC. Integrates the functionality of Kerberos KDC, GSS acceptor and EAP authenticator.

- AAA server. Integrates the functionality of the EAP server.

- Identity Provider. Generates authentication statements upon a request from the AAA server and attribute statements upon a request from the KDC.

- Policy Decision Point (PDP). Manages the set of access control policies for the domain of the KDC and takes authorization decisions based on the provided information.

- MetaData Service (MDS). Maintains a database of every service metadata within the federation. The MDS can be queried by any member of the federation to obtain information about other member's public service [OASIS.saml-metadata-2.0-os].

- Application Server. Provides a service valuable to the End User, whose access is controlled by means of the Kerberos protocol.

3. Operation

3.1. Discovery of the KDC

In federated environments, where users move between different organizations, the specific location of the KDC in any visited domain should by dynamically discovered by the End User. This may be achieved by the use of DNS SRV RR queries as defined in [RFC4120] or the DHCPv6 protocol as defined in [I-D.sakane-dhc-dhcpv6-kdc-option].

3.2. Pre-authentication with the KDC

Once the KDC location is known by the End User, the pre-authentication process is performed as depicted (in a very simplified way) in Figure 1. A detailed description of these steps is provided following.

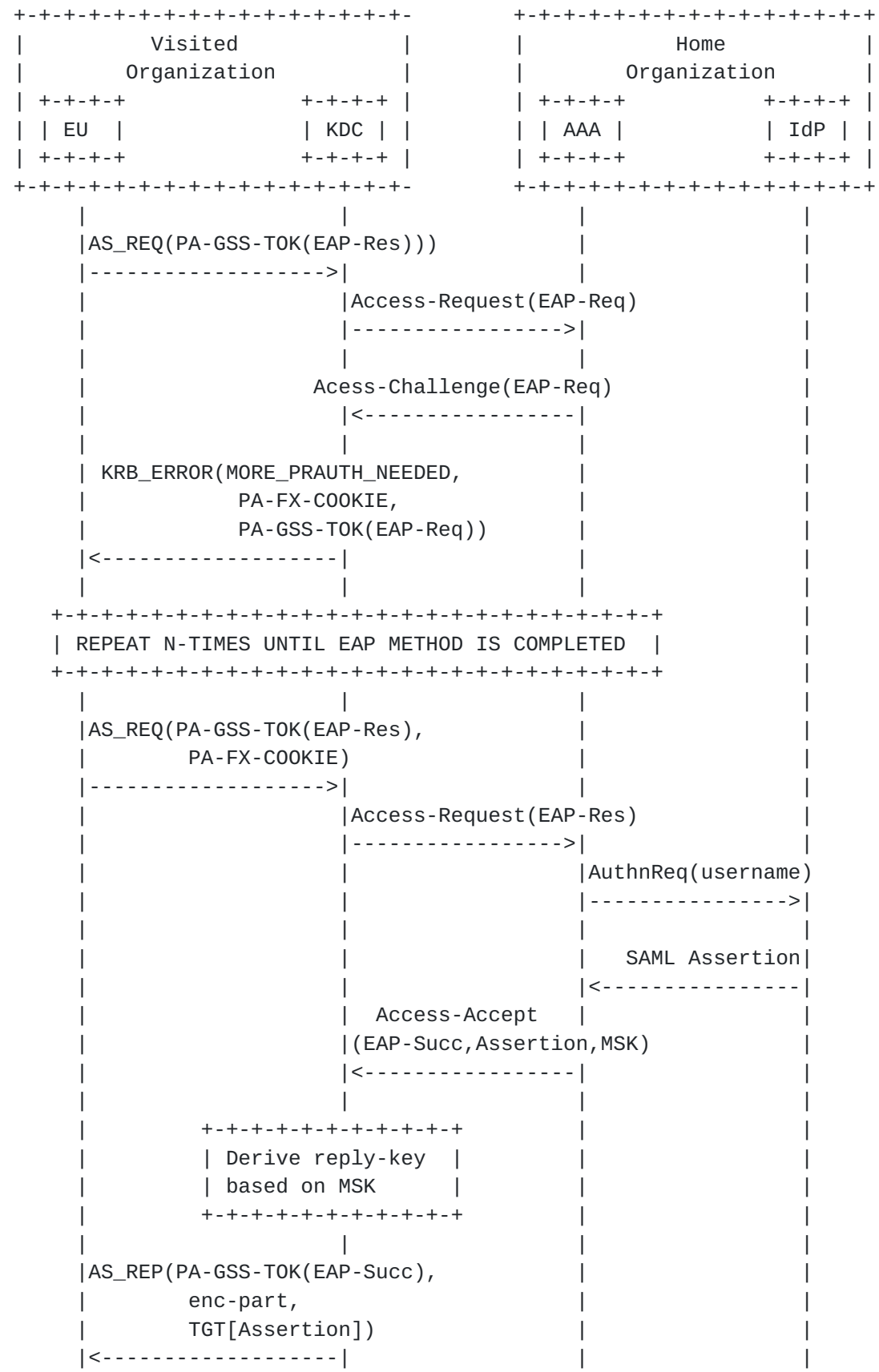
Visited Organization
EU
KDC
Home Organization
AAA
IdP
AS_REQ(PA-GSS-TOK(EAP-Res)))
Access-Request(EAP-Req)
Acess-Challenge(EAP-Req)
KRB_ERROR(MORE_PRAUTH_NEEDED,
PA-FX-COOKIE,
PA-GSS-TOK(EAP-Req))
REPEAT N-TIMES UNTIL EAP METHOD IS COMPLETED
AS_REQ(PA-GSS-TOK(EAP-Res),
PA-FX-COOKIE)
Access-Request(EAP-Res)
AuthnReq(username)
SAML Assertion
Access-Accept
(EAP-Succ,Assertion,MSK)
Derive reply-key
based on MSK
AS_REP(PA-GSS-TOK(EAP-Succ),
enc-part,
TGT[Assertion])

Figure 1: EAP-GSS pre-authentication with the KDC

1. Kerberos client calls to GSS_Init_sec_ctx() to obtain the a GSS token to be sent to the KDC. Details on this process are described in [I-D.perez-krb-wg-gss-preauth]. The selected GSS mechanism is GSS-EAP, defined in [I-D.ietf-abfab-gss-eap].

2. The GSS_Init_sec_ctx() call is treated by the GSS-EAP mechanism, which generates the actual GSS token following the specifications provided in [I-D.ietf-abfab-gss-eap]. Usually, this GSS token will contain an EAP response message. At this level, the GSS-EAP mechanism will use the EAP identity of the End User.

3. Once the Kerberos client has obtained the GSS token, it is encapsulated into a PA-GSS-TOKEN pre-authentication data element and included into the KRB_AS_REQ message (as specified in [I-D.perez-krb-wg-gss-preauth]). The user may belong to a different organization than the KDC and, therefore, its client name may not be found in the KDC's local database. To avoid the generation of an error of type KDC_ERR_C_PRINCIPAL_UNKNOWN, the Kerberos client sets the cname field of the KRB_AS_REQ message to a new fixed value, WELLKNOWN:FEDERATED, following the model proposed in [RFC6111]. In this manner, the KDC is warned that the End User belongs to the federation and not local verification of credentials should be done.

4. On the reception of the KRB_AS_REQ message, the KDC omits the local database lookup of the client name, since WELLKNOWN: FEDERATED is indicated. Then, the KDC calls the GSS_Accept_sec_ctx() function, as described in [I-D.perez-krb-wg-gss-preauth], to process the GSS token received in the PA-GSS-TOKEN pre-authentication data element.

5. The GSS_Accept_sec_ctx() call is treated by the GSS-EAP mechanism, which extracts EAP packet contained on it, determines the AAA server where it should be forwarded and encapsulates it into the proper AAA protocol (i.e. RADIUS or Diameter), as described in [I-D.ietf-abfab-gss-eap].

6. The AAA server processes the EAP packet and generates a new EAP Request for the End User. If the response is an EAP Success, the process continues in step 10. Otherwise, the AAA server encapsulates the EAP packet into a AAA message and sends it to the KDC. Note: For simplicity, AAA proxies are not considered. More details are provided in [I-D.ietf-abfab-gss-eap].

7. The GSS-EAP mechanism in the KDC processes the AAA message and generates a new GSS token with the obtained EAP request.

8. As a result of the call to the GSS_Accept_sec_ctx() function, the KDC obtains the GSS token, which is encapsulated into a new PA-GSS-TOKEN element and sent to the Kerberos client into a KRB_ERR message with code MORE_PREAUTH_DATA_REQUIRED as specified in [I-D.perez-krb-wg-gss-preauth]. This KRB_ERR message also contains a PA-FX-COOKIE element where the KDC introduces the value of the context handle obtained after the call.

9. The Kerberos client processes the KRB_ERR message, obtains the GSS token and calls to the GSS_Init_sec_ctx() function. The process continues as defined in step 1.

10. If the AAA server determines that the authentication has been completed, before sending the proper message to the KDC it contacts with its local IdP to obtain an SAML Authentication Statement. This is accomplished by sending a SAML AuthnRequest message to the IdP, indicating the EAP identity as the value of the Subject. This message is generated following the SOAP-based authentication profile described in [LIBERTY.idwsf-authn-svc-v2.0]. Alternatively, the IdP could be collocated with the AAA server. In this case, this and the next step would be omitted as the AAA server could issue the SAML Assertion by itself. Another option would be that the KDC generates the SAML AuthnRequest instead of the AAA server, since it will be the ultimate consumer of the produced Assertion.

11. The IdP authenticates the user based on the trust established in the organization between the AAA server (acting as the attester) and the IdP, and generates a SAML Assertion in response. This Assertion, which contains an Authentication Statement, will be used to refer to this authentication process in the future (e.g. to request attributes).

12. The AAA server verifies the Authentication Statement and encapsulates the Assertion into an SAML-AAA attribute (e.g. following the [I-D.howlett-radius-saml-attr]. Then it sends the AAA response message to the KDC including this attribute along with the EAP Success packet and the derived MSK.

13. The GSS-EAP mechanism processes the AAA response as described in [I-D.ietf-abfab-gss-eap]. The EAP identity is exported as initiator name. This name will also include the received Assertion as an attribute, following the specifications in GSS-naming [I-D.ietf-kitten-gssapi-naming-exts]. The MSK is used to

derive the GMSK.

14. The KDC obtains, as a result of the last call to the GSS_Accept_sec_ctx(), the final GSS token, and the initiator name. As described in [I-D.perez-krb-wg-gss-preauth], the token is encapsulated into a PA-GSS-TOKEN, while the initiator name is used as the cname field in both, the KRB_AS_REP message and the TGT. Furthermore, the KDC obtains, using the GSS_Get_name_attribute(), the SAML Assertion. This Assertion is included in a new authorization data element (ADE) into the authorization_data field of the TGT. By this way, the TGS will receive the Assertion in the KRB_TGS_REQ. Following the specifications in [I-D.perez-krb-wg-gss-preauth], the KDC uses the GSS_Pseudo_random() call to generate the reply key with which encrypting the enc-part of the KRB_AS_REP message (Replacing-reply-key facility)

15. Finally, the Kerberos client processes the KRB_AS_REP message. As a first step, the PA-GSS-TOKEN is processed as specified in [I-D.perez-krb-wg-gss-preauth]. The GSS-EAP mechanism extracts the EAP packet from the token and derives the MSK and the GMSK. The Kerberos client uses the GSS_Pseudo_random() call to generate the reply key, and decrypts the enc-part of the KRB_AS_REP message. After that, the Kerberos client stores the TGT into the credentials cache associated to the identity indicated in the cname field.

3.3. Authorization

With the TGT, the End User can request Service Tickets (ST) for the different services in the domain by means of a TGS exchange. In this process, the TGS can take an authorization decision based on the identity information of the End User, thanks to the authorization information (i.e. Assertion) the AS included in the TGT. Both the service and the client are completely agnostic of this authorization process, thus they do not require changes. A simplification of the process is outlined in Figure 2. A detailed description is provided below.

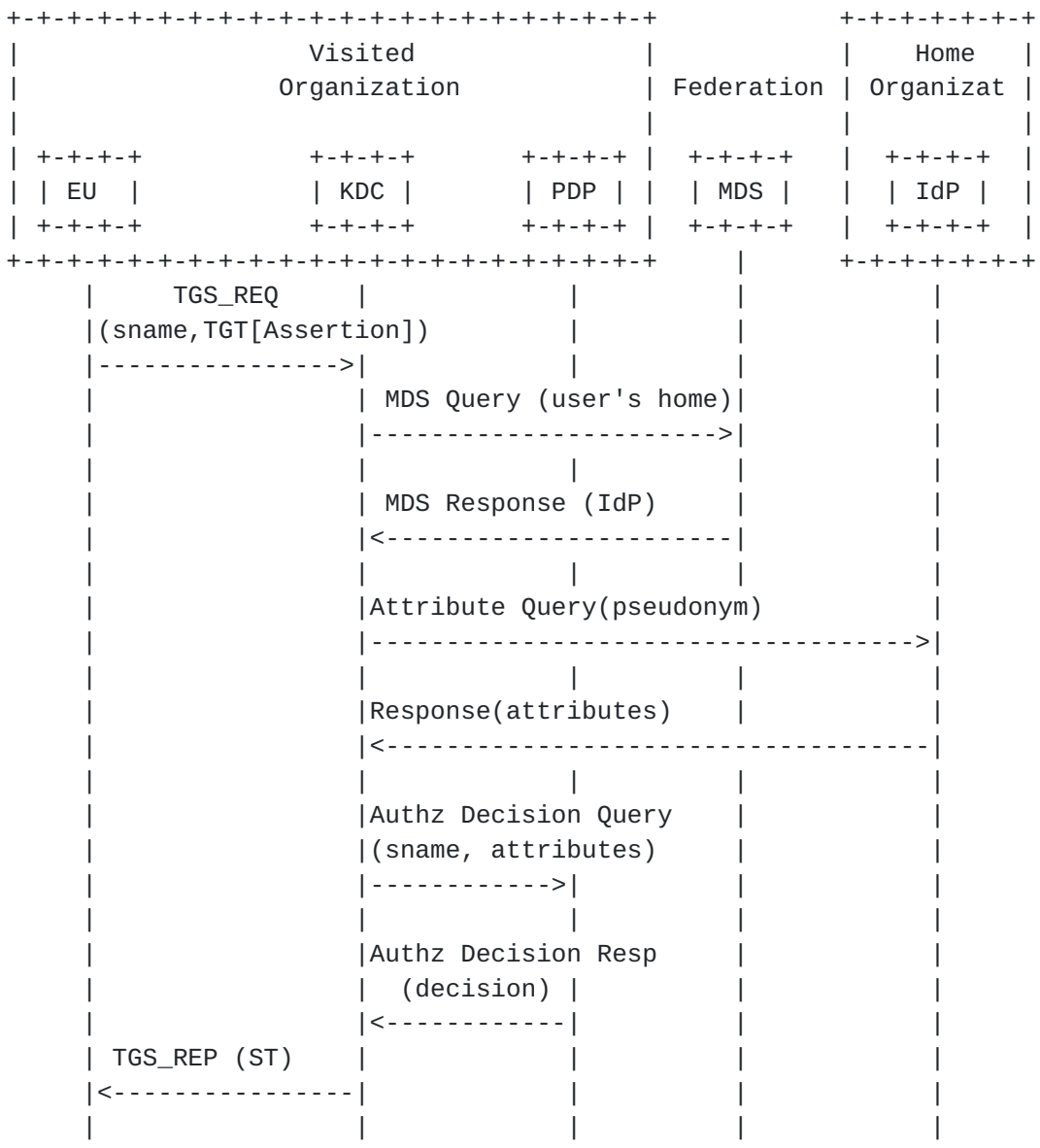

Figure 2: Authorization process

1. The Kerberos client creates a new KRB_TGS_REQ message, following what is specified in [RFC4120]. This message transports the TGT obtained during the authentication phase.

2. The TGS (KDC) processes the KRB_TGS_REQ message as usual. When the TGS processes the authorization data element (ADE) containing the Assertion, it contacts with the federation's MDS to obtain the location of the End User's home IdP. This information is required in order to contact with the End User's IdP to request some user attributes to perform the authorization decision.

3. With the metadata information the TGS is able to contact with the End User's IdP directly. This contact is performed by means of a SAML Attribute Query message [OASIS.saml-core-2.0-os], indicating the pseudonym of the user included in the assertion as the Subject.

4. The IdP recognises the pseudonym and provides the requested attributes in a SAML Response containing one or more SAML Attribute Statements.

5. The TGS provides the gathered attributes to the local PDP, along with information about the resource (sname) and the action to be performed, using an XACML AuthzDecision Query message. With this information the PDP queries its local policy database and takes an authorization decision. This decision is provided to the TGS using a XACML AuthzDecision Response message.

6. If the decision is PERMIT, the TGS issues the ticket for the requested service. Otherwise, if the decision is DENY, the client is provided with a Kerberos error message and the ST is not delivered.

There is one way of simplifying the authorization work-flow by allowing IdPs to return SAML Attribute Statements during the first step of the authorization phase. In this case, the AAA server will receive a SAML Attribute Statement holding the end user attributes along with the Authentication Statement in the Assertion. This way the MDS Query and the Attribute Query exchanges could be omitted, since the Assertion already contains the required information. However, this approach does not allow the IdP to discriminate what attributes should be returned based on the preferred services, because they are known in the KRB_TGS exchange.

3.4. Access to the Application Service

With the ST, the user can access the Application Service using standard Kerberos, as described in [RFC4120]

4. Security Considerations

4.1. AAA/EAP key management in Kerberos

The use of EAP for Kerberos pre-authentication has security implications, specially in key distribution and management. The security analysis described in [RFC4962] and [RFC5247] are applicable here. Indeed, intermediate AAA proxies placed between the KDC and the home AAA server can observe the distributed MSK that will be used

to derive the Kerberos secret key.

However, the trust model in federated environments assumes that intermediate AAA proxies are trusted entities. Moreover, as [RFC4962] explains, some key wrapping techniques can be applied to provide confidentiality, integrity and replay protection to the distributed key material between each pair of AAA entities (e.g. AAA proxies).

4.2. Trust relationships

This work assumes the existence of a transitive trust relationship for authentication between the involved domains thanks to the deployed AAA infrastructure. Besides, regarding the authorization process, it is generally assumed the use of a direct trust relationship, allowing the protection of SAML messages directly between domains (step 2 and 3). Usually PKI architecture are used to deploy this trust. Following this approach IdPs and KDCs should be fed with cryptographic material.

Nevertheless, other approach can be followed to avoid the deployment of an additional infrastructure (e.g. PKI). We could leverage the transitive trust relationship defined by the deployed AAA infrastructure to perform safely the attribute recovery process by means of the AAA protocol. In this case, it is necessary to define new extensions to standard AAA protocols.

5. IANA Considerations

This document has no actions for IANA.

6. Normative References

[I-D.howlett-radius-saml-attr]
Howlett, J., "A RADIUS attribute for SAML constructs", draft-howlett-radius-saml-attr-00 (work in progress), May 2010.

[I-D.ietf-abfab-gss-eap]
Hartman, S. and J. Howlett, "A GSS-API Mechanism for the Extensible Authentication Protocol", draft-ietf-abfab-gss-eap-01 (work in progress), February 2011.

[I-D.ietf-kitten-gssapi-naming-exts]
Williams, N., Johansson, L., Hartman, S., and S.

Josefsson, "GSS-API Naming Extensions", draft-ietf-kitten-gssapi-naming-exts-11 (work in progress), May 2011.

[I-D.perez-krb-wg-gss-preauth]
Perez-Mendez, A., Lopez, R., Pereniguez-Garcia, F., and G. Lopez-Millan, "GSS-API pre-authentication for Kerberos", draft-perez-krb-wg-gss-preauth-00 (work in progress), May 2011.

[I-D.sakane-dhc-dhcpv6-kdc-option]
Sakane, S. and M. Ishiyama, "Kerberos Options for DHCPv6", draft-sakane-dhc-dhcpv6-kdc-option-11 (work in progress), May 2011.

[LIBERTY.idwsf-authn-svc-v2.0]
Hodges, J., Aarts, R., Madsen, P., and S. Cantor, "Liberty ID-WSF Authentication, Single Sign-On, and Identity Mapping Services Specification", Liberty Alliance liberty-idwsf-authn-svc-v2.0, 2006.

[OASIS.saml-core-2.0-os]
Cantor, S., Kemp, J., Philpott, R., and E. Maler, "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard saml-core-2.0-os, March 2005.

[OASIS.saml-metadata-2.0-os]
Cantor, S., Moreh, J., Philpott, R., and E. Maler, "Metadata for the Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard saml-metadata-2.0-os, March 2005.

[OASIS.xacml-2.0-core-spec-os]
Moses, T., "eXtensible Access Control Markup Language (XACML) Version 2.0", OASIS
Standard xacml-2.0-core-spec-os, February 2005.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[RFC4962] Housley, R. and B. Aboba, "Guidance for Authentication, Authorization, and Accounting (AAA) Key Management", BCP 132, RFC 4962, July 2007.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, August 2008.

[RFC6111] Zhu, L., "Additional Kerberos Naming Constraints", RFC 6111, April 2011.

Authors' Addresses

Alejandro Perez-Mendez (Ed.)
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

Fernando Pereniguez-Garcia
University of Murcia
Campus de Espinardo S/N, Faculty of Computer Science
Murcia, 30100
Spain

Phone: +34 868 88 78 82
Email: pereniguez@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